Csdn.net 技术改变世界 创新驱动中国

Programmer 程序员

2009年 01月刊

邮发代号:2-665 定价:10元

产品经理的5种能力

Business of software已成为IT企业重中之重，而产品经理这一角色，既代表了软件在商业社会中的产品价值观，也是开发者的另一条职业发展之路。本期封面报道，告诉您如何成功经营一个产品！

■ 高端视点

危机，也是企业走向强大的机会

精彩推荐 ■

大型Web 2.0网站架构纵横谈

敏捷外包的14条原则

Perl 6的未来

放大愉悦

——《征途》策划新理念

一分钟先生：如何做时间管理?

■ 月度关注

磨刀不误砍柴工

——IDE助你提高开发效率

时至今日，IDE已陪伴我们走过44年的风风雨雨。这些年里，它不断影响着我们的开发过程，提高着我们的开发速度。让我们走进Java IDE的世界，体验技术进步给我们带来的便捷。

ISSN 1672-3252

9 771672 325098 01>

程序员 Programmer 2009年 01月刊 技术改变世界 创新驱动中国

特别感谢 Contributors

（排名不分先后，以姓氏首字母为序）

陈明杰

感谢傲游公司 CEO 陈明杰接受本刊专访，详见本期封面报道。

成修治

感谢红旗 2000 开源软件技术部总监成修治先生分享观点，详见本期高端视点。

傅盛

感谢经纬中国投资副总裁傅盛接受本刊专访，详见本期封面报道。

Dennis Flanagan

感谢微软公司 Windows 产品总经理 Dennis Flanagan 先生接受本刊采访，详见本期封面报道。

高焕堂

感谢台湾知名软件架构设计大师高焕堂先生对本刊的一贯支持。

韩少云

感谢达内科技 CEO 韩少云先生接受本刊采访，详见“报道”栏目。

Vikas Hazrati

感谢 Vikas Hazrati 分享他关于敏捷和外包的思考，详见本期“实践”栏目。

姜太文

感谢开源社区 XOOPS 项目负责人姜太文为本刊撰文，详见本期封面报道。

蒋清野

感谢 Sun 中国技术社区高级经理蒋清野先生与本刊分享对 OpenSolaris 的见解 ，详见“评论”栏目。

焦烈焱

感谢普元软件首席架构师焦烈焱接受本刊专访，详见本期封面报道。

Ivar Jacobson

感谢软件工程之父、雅各布森图际股份有限公司董事长 Ivar Jacobson 先生对本刊的一贯支持。

刘刚

感谢瑞星研发副总经理刘刚接受本刊记者的采访，分享关于互联网安全的观点，详见本期报道。

马占凯

感谢搜狐公司产品经理马占凯接受本刊专访，详见本期封面报道。

苗峰

感谢北京用友华表软件技术有限公司总经理苗峰先生分享对业界的思考，详见本期高端视点。

牛新庄

感谢 IBM 官方资深培训讲师牛新庄博士分享自己学习数据库的经验，详见本期实践栏目。

David Nicolette

感谢资深敏捷专家 David Nicolette 为本刊分享他关于敏捷的所思所想，详见本期“实践”栏目。

Gregor Roth

感谢 xLightweb HTTP 程序库的作者、软件架构师 Gregor Roth 为本刊分享关于服务负载平衡架构方面的精彩内容，详见本期“架构”栏目。

孙丕恕

感谢浪潮集团董事长兼首席执行官孙丕恕先生分享对金融危机的看法，详见本期高端视点。

吴穹

感谢雅各布森国际股份有限公司中国公司董事总经理吴穹博士对本刊的一贯支持。

谢克人

感谢新锐集团有限公司董事长谢克人先生为本刊分享有关成长型企业管理的经验，详见本期高端视点。

徐峰

感谢软件工程专家、中国系统分析顾问团软件工程首席顾问徐峰先生，与本刊分享有关需求获取的宝贵经验，详见本期“实践”栏目。

严孟宇

感谢顶点软件 CEO 严孟宇接受本刊专访，详见本期封面报道。

张银奎

感谢英特尔亚太研究中心高级软件工程师张银奎先生为本刊主持调试之剑栏目。

周铁人

感谢 TechExcel 公司 CEO 周铁人接受本刊专访，详见本期封面报道。

卷首语

Foreword

丢掉幻想，准备“山寨”

压缩预算、冻结招聘、降薪、裁员、倒闭，最近听到了不少这样的坏消息，中国经济无疑是真的遇到了困难。而今天世界经济面临的问题，是自布雷顿森林体系解体以来，已经过三十多年的能量积累，并且在最近二十年里，被苏东巨变、中国崛起、科技狂飙、房产泡沫等一系列事件逐级放大，如今一旦崩塌，又怎么可能是小打小闹？又怎么会像很多人希望的那样“半年转阴，一年放晴”？

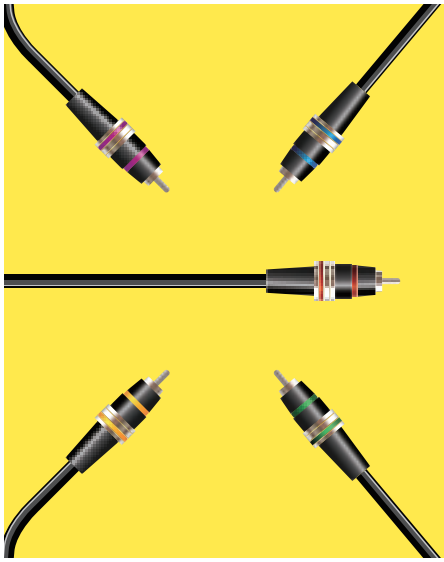
人类社会有一种倾向，一旦草莽搏杀的淘金热过去，“真本领”就变得次要了。西装革履的家伙们会主动跑过来帮忙构造上层建筑，那里有无懈可击的资本制度，有精美漂亮的层级结构，有酒精考验的人际关系，更有让他们如鱼得水的办公室政治。过去的十几年里，中国软件产业从无到有，互联网从零到世界第一，取得了巨大的进步，同时也建立起这样的一整套的上层建筑。我们痴迷地追逐名牌大学的MBA和五百强公司的职位，拉出一大票唬人的架构、模式和方法学来装点门面，努力提高自己夹杂中文的英语，学习社交礼仪和职场政治……这不是在讥讽或者批评什么人，这些做法只是我们为了适应秩序所做的必然选择。然而，如果一个新的、秩序重建的时期到来，那么我们也应该改变自己的观念和行为，建立新的气质和精神。

我相信这样的精神已经存在了，那就是所谓“山寨”精神，它体现在一年来层出不穷的民间“山寨事件”中。从无敌霸王手机到“百谷虎超级搜索引擎”，从《红楼梦》、《西游记》、《百家讲坛》到民间“春晚”，阅人无数，“雷”人无数。也许有人看到的是“山寨版”无耻的抄袭和恶搞，但我看到的却是大胆的拿来主义，源于百姓、用于百姓的平民主义，张扬而质朴的个性，对于装模作样的既有秩序的藐视与颠覆，以及，最重要的，久违的中国人的创造力和想象力。这些恣意横生的山寨作品，我们那些满脑子条条框框模式思想的架构师们做得出来吗？我们那些整天开口闭口美国如何，却对自己的父老乡亲一无所知的“大师”们做得出来吗？

变局之下，“山寨”精神尤显重要。中国未来的发展之路，将会以提振内需为重要手段。软件本来就是最能够发挥本土优势、“山寨”精神的领域，“四万亿”、“核高基”等一系列国家计划的提出，给中国软件自主创新创造了前所未有的历史机遇。我看一百多年来中国凡是做得好的事情，无不是“山寨”精神的体现，而做得差的事情，则有不少是食洋不化的结果。我相信，假使若干年后，中国软件得以因其创新性和在中国经济中的作用而令世人称道，那一定也是有“山寨”精神在其中的作用。

经济的危机要来啦，这是坏事，也是好事。从好的方面来说，十几年来构筑的虚浮的东西已经够高够重，把它们撕掉烧掉，回到“山寨”好了。“山寨”当然不是终点，而只是一个既有意思又有启示的起点。相信凭着“山寨”精神，我们中国的程序员终将造出“雷”倒微软、Oracle和Linus Torvalds的一流软件。

Programmer 程序员 CONTENTS
技术改变世界 创新驱动中国 ● http://www.programmer.com.cn



封面报道 Cover Story

P32 产品经理的5种能力

Business of software已成为IT企业重中之重，而产品经理这一角色，既代表了软件在商业社会中的产品价值观，也反映出开发者的另一条职业通道。本期封面报道，讲述如何成功经营一个产品！

32 产品经理的五项能力
36 360安全卫士规划故事
38 XOOPS发布有期：谁说开源不计划?
40 TechExcel的“灵魂缔造者”
42 LiveBOS的深度需求工程
44 创意搜狗输入法设计
46 浅谈51.com产品设计
48 傲游的设计思想之源
50 普元：执行力打造企业级平台产品
52 市场与Windows的双向选择

固定专栏 Columns

9 名人堂 Hall of Fame
14 业界新闻 News
15 程序天下事 Technical News
31 声音与幽默 Humor

高端视点 Leaders' Viewpoint

10 危机，也是企业走向强大的机会
11 成长型企业如何成功?
12 也谈中国信息化长尾市场

13 如何与开源社区共发展

报道 Report

23 雪中送炭，王者归来
——2008软件开发2.0技术大会后记
26 全民动员，保证互联网安全
27 寒冬中的远见
——达内科技暖冬工程
28 最佳实践经验助力软件企业发展
29 洞察危机中之机遇，把握自身命运

架构 Architecture

54 大型Web 2.0网站架构纵横谈（一）
随着Web 2.0的概念逐渐深入人心，相关应用也越来越多，众多大型Web 2.0网站在架构上面临着全新的挑战和问题。作为该系列的第一篇文章，本文先从“数据”和“安全”两个话题谈起。
57 负载均衡架构系列2

实践 Practices

61 敏捷外包的14条原则
突如其来的金融海啸，使得全球的软件外包产业不得不重新洗牌；这对中国软件外包企业可谓是千载难逢的机会。本文提出的基于敏捷的软件外包原则，希望能够给我们的软件人一些建议和思考。
66 小型软件公司绩效考核
国内有着诸多小型软件公司，“如何做绩效考核”是这些公司普遍面临的问题。本文从“组织模式”、“团队建设”、“开发流程”等几个方面探讨一些行之有效的绩效考核方法。
68 活灵活现用GIT——基础篇
在进行源代码版本管理时，继Subversion之后，GIT成为软件开发人员的新好。本文首先介绍Git的基本概念和原理，并穿插一些常用命令的使用。
70 需求沟通中的艺术
在需求获取过程中，讲究艺术技巧的沟通方式

P23 报道

雪中送炭，王者归来
——2008 软件开发 2.0 技术大会后记

可以带领大家跨越甲方和乙方的鸿沟，抓住问题的本质，为项目成功奠定坚实基础。文中的几个故事和技巧就明确体现了这一点。

72 六西格玛和软件开发（2）
继上期文章之后，本文首先从科学严谨性的角度对六西格玛进行了分析，接下来分析指出严谨的科学方法并不一定适用于所有的决策。

78 如何做时间管理?

80 牛新庄：我的数据库学习“曲线”

技术 Technology

83 C++特性约束检查（下）
本文讨论了一种允许用户定义任意代码特性集，并保证在编译时被调用函数满足调用者所有代码特性要求的机制。

87 基于Linux模块的防火墙系统
本文首先分析了Linux的Netfilter的内核架构，在此基础上，采用模块编程方式开发了一个高效实用的包过滤型防火墙系统。

90 Linux网卡驱动开发一例

94 Open API分析与实践
Open API是近一年来比较流行的互联网技术，本文作者结合自己的经验，讨论了Open API的理念和实践。

100 图像的复杂度及应用

102 Perl 6的未来

105 Perl在生物研究中的应用
实验生物学家通常要不断地面对大量的文件，这些文件体积硕大、格式不相兼容，如何解决？本文作者使用Perl，对此领域问题提供了应对之道。

108 放大愉悦——《征途》策划理念

调试之剑 Debugging Sword

111 步步为营
——如何调试操作系统加载阶段的故障

月度关注 Monthly Focus

114 集成开发环境简史

117 使用Mylyn提高开发效率

119 IntelliJ IDEA：开发人员利器

产品推荐 Products Recommendation

122 新产品新工具

124 开源项目

125 创新产品

126 Geek产品

图书 Books

128 编译领域里程碑之作
——龙书《编译原理》

129 成功演示六要素

131 新书上架

评论 Comments

133 回顾：OpenSolaris 2008.11

135 JAVA与数字电视：天堂还是地狱?

P126

Contents

32 5 capabilities of product manager

Nowadays, business of software is more and more important for developers and programmers. A lot of them select their occupational planning to be a product manager which plays the role of represents the value of commercial society. In this issue, we will discuss how to be a product manager, and what basis of capabilities you need, to become a product manager, even how to manage your software products.

61 Agile Offshoring : It's hard work but it works!

Software Off-shoring is a reality of the day however there are many projects which fail due to incorrect off-shoring. Apart from tremendous advantages, off-shoring brings additional complexity, risk and avenues for wastage. This experience report will discuss how we turned off-shoring into a successful model based on Toyota manufacturing Process. We call this methodology 'Lean Agile Off-shoring.'

66 Performance Appraisal for Small Software Companies

Performance appraisal is a common problem for many Chinese small software companies. This paper suggests some effective methods from the aspects like organization patters, team building and development process.

96 Analyses on Open API

Open API is hot topic in recent one year, the author of this article discussed experience about using Open API.

114 Brothers do not misuse wood workers——How IDE boost up you develop

44 years before, Dartmouth BASIC IDE released which is the first IDE of the world.In this 44 years, IDEs have changed the development so much.Although somebody says: IDEs let programmers become more lazy, we can not write off IDE's contribution.As the result, "Monthly Review" focus on Java IDE, and hope you enjoy you life with new tech.

Programmer 程序员

2009年1月刊 总第195期

主管：中国社会科学院
主办：中国社会科学院文献信息中心
出版：《程序员》杂志社
网址： http://www.programmer.com.cn
国际刊号：ISSN 1672-3252
国内刊号：CN11-5038/G2
邮发代号：2-665
广告经营许可证号：京东工商广字0188号

总编：黄长著 Editor-in-chief：Huang Changzhu
社长/常务副总编：张悦校 President：Zhang Yuexiao
副社长：蒋 涛 Vice President：Jiang Tao
编委会：黄长著 张悦校 陈洋彬 蒋涛 曾登高 韩磊
Editorial Member：Huang Changzhu Zhang Yuexiao Chen Yangbin Jiang Tao Zeng Denggao Han Lei

执行主编：孟迎霞 Executive Editor-in-chief：Meng Yingxia
技术主编：孟岩 Editor-in-chief（Technical）：Meng Yan
编辑部主任：孟迎霞（兼） Director：Meng Yingxia
编辑部副主任：欧阳璟 Deputy Director：Ouyang Jing
责任编辑：郑柯 赵健平 李雨来 彭一凡
Editors：Zheng Ke Zhao Jianping Li Yulai Peng Yifan
特邀编辑：方梁 高昂 常政
Contributing Editors：Fang Liang Gao Aang Chang Zheng
美术总监：张浩祥 Art Director：Zhang Haoxiang
美术编辑：吴志民 Art Editor：Wu Zhimin
Tel：010-64351458
Email：editor@csdn.net

发行部 Distribution Dept.010-64351431
Email：sales@csdn.net

广告总代理：北京创新乐知广告有限公司
Sole Advertising Agency：Bejing CSDN Co.,Ltd
Tel：010-64376055
Email：ad@csdn.net
Marketing Dept：010-51661202（ext 149）
Email：market@csdn.net

读者服务部
Readers service Dept.
网上订购：www.dearbook.com
读者信箱：reader@csdn.net
地址：北京市朝阳区酒仙桥路14号兆维工业园B区3楼2门1层
Address: B3-2-1F, Zhaowei Industry Park, No. 14 Jiuxianqiao Road, Chaoyang Dist, Beijing
邮政编码：100016
电话：010-64351425
传真：010-64348545

法律顾问：北京鸿盛律师事务所 王杰
Law Consultant：Beijng Hengsheng Lawyer Firm
印刷：北京盛通印刷股份有限公司
Print: Beijing Shengtong Printing Co., Ltd.
出版日期：每月1日
Publication Date：the first day per month
零售价：RMB 10.00元 新台币 260元 HK $ 25.00（港、澳）
US $ 6.00（海外）
Retail Price：RMB 10, NT$260, HK $ 25.00,US $ 6.00

一代鬼才，COM之王
——Don Box

■ 文 / 吕娜

Microsoft之前的.Net平台演变有两个时代——DOS时代和COM时代。在这九年的整个COM时代，对于开发分布式和并发式系统而言，COM是应用最为广泛的一种对象类型，如果在Window或NT上建立系统，那么就不可能躲开COM，本期名人堂栏目的主人公就是COM领域的泰斗Don Box。

1983年，早在Don Box还是一名数学系的大学生的时候，就学会了编程，开始了他的计算机技术生涯。1994年，在俄勒冈学院执教期间，与加利福尼亚大学教书的好友Andrew Harrison合作一门课程Essential Windows时，开始在COM世界摸索前进。同年8月，Don Box编写了Essential COM课程，并在1998年出版了此后影响深远的著作《Essential COM》。Don Box成为了一位著名的教育家，是软件集成技术方面的权威，被公认为组件对象模型（COM）领域的权威人物。

而后Don Box与他人共同创立了一家叫Develop Mentor的公司，在COM方面找到了很多乐趣。这家公司主要为软件工业提供教育和支持，是一家组件软件思想库，致力于为开发人员提供COM、Java和XML使用方面的培训。作为Develop Mentor公司的合作创建人，Don Box曾提供过基于COM的项目咨询。

在上世纪90年代，他花费大量时间为CLR积极筹建组件对象模型[Component Object Mode(COM)]社区。之后他与Bob Atkinson,Mohsen Al-Ghosein, 及 Dave Winer 缔 造了SOAP（Simple Object Access Protocol，简单对象访问协议）规范，并成为W3C Schema工作组成员。尽管多年以来，Don Box一直在努力让SOAP成为程序在Internet上通讯的一种标准方法，但结果不尽如人意，因为他发现原始的SOAP和XML与他现在正在使用的所有编译器不一致。

Don Box拥有深邃的技术洞察力，是出了名的鬼才，他的研究领域包括组件软件集成、并行编程、基于XML的串行化和元数据协议。关于COM，没有任何人能阐释得比Don Box更棒。他是畅销技术专著《Essential COM》、《Effective COM》和《Essential XML》的作者及合著者。Don Box著作最吸引人的地方在于说明COM时的组织结构，他善于“用COM来解决常见问题”，先构建开发分布式和并发式系统时所遇到的各种问题，然后向你展示COM是如何解决这些问题的。因此，即使你对COM一无所知，也会被其中清晰而又简单的概念模型所导引，直至找到理解问题之症结，及赋予COM结构和行为的力量之所在。

正如Microsoft公司首席程序经理James S.Miller说：“Don是我所见到的最好的教师：他不仅能够清晰地讲解，很好地辅以实验，写作能力也无可挑剔，而且，他还能够讲解得非常细致透彻，从宏观问题‘为什么以那种方式构建这个体系结构？’，到细节问题‘那个bit到底是在什么位置？’。就这样，Don教会我（连同其他成千上万的技术人员）什么是COM，为什么要使用COM，COM有什么问题。考虑到我们的实际工作，他还即兴将COM技术同CORBA技术，以及我们正在设计与构建的Web技术做了令人惊讶的比较。”

除了在Develop Mentor公司的工作外，Don还曾为Microsoft Systems Journal的特约编辑及专栏作家（后来的MSDN杂志），为Security Q&A专栏撰稿。此外，Box还负责创建了一个时尚栏目“与Don Boxers同行”。同时他也是Addison Wesley的编辑，并成功发起过两次Microsoft developer audience。2001年，Don Boxers作为软件架构师加入了Microsoft .NET Developer and Platform Evangelism Group。

自1993年Windows NT3.1发布为标志开始，微软进入COM时代，2002年公共语言运行库[Common Language Runtime(CLR)]作为.NET Framework的一部分发布，COM时代终结。1993-2002的九年间，COM经历了重大的变化，如此经久，证明其魅力所在。也许就像他发明的那个词一样——“COM，我的爱（COM is love）”。■

危机，也是企业走向强大的机会

当前世界经济正处在一个空前紧张的时期，金融危机带来的经济危机对世界经济的影响正在逐步显现出来。IT产业可能也难逃厄运，一批国际IT业巨头开始裁员，并随之相应调低年度营收预期。与欧美相比，由于我国IT行业产品和消费结构性的差异，再加上我国IT行业自身受宏观虚拟经济的影响有限，所以到目前为止，此次国际金融危机对我国IT行业的影响并不十分明显。

但是，这场危机肯定会对国内的IT企业造成影响。具体来说主要是两个方面，一方面是IT投资的积极性在减少，在IT方面的投入会更加谨慎；另一方面，由于全球政治家、企业家都把目光聚集到中国市场，国内厂商也会因为自身原有目标市场受到冲击而加紧进入竞争对手的优势行业。因此2009年中国市场的竞争会更加激烈。

艰难之时往往也就是机遇之时，我们希望在此次危机中通过一些措施，不仅能够安然度过危机，并且能够把握住一些机会，让自己走向强大。具体而言，我们主要从以下几个方面来着手应对：

推进产业结构升级，依靠自主创新，向IT产业链上游拓展

中国IT产业长期处于全球IT产业链的中下游环节。通过产业升级，向产业链上游延伸，在全球IT产业竞争中争得更大的话语权，是我们抵御困难与风险的主要途径。同时，需要进一步提高行业软件和企业软件应用中间件的研发能力和水平，从而提高软件产业的效率，实现软件领域向上游的拓展。

精耕细作国内市场，不断地对国内市场进一步细分化拓展。

在金融危机的背景下，企业更需要通过加强管理、节省成本来提升自身竞争力；也应该重新审视自己的业务模式、业务流程以及信息系统。这会出现一些新的市场机会，同时也是IT企业深入了解客户需求、对市场进行进一步细分、调整自己的产品结构、提升自身服务能力和竞争力的机会。

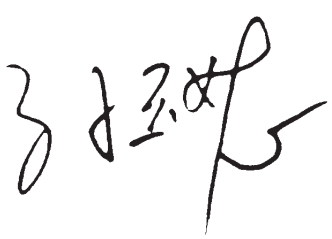
实施有“竞”有“合”的全球化战略，注重向发展中国家进行产业转移

我们应该抓住跨国公司向中国深入拓展的机会，加大国际合作力度，提升技术、管理、国际运作能力。同时，国内的企业应注重向发展中国家进行产业转移。这些国家虽然法制政策等环境还并不完善，却有着非常多的市场机会。

积极招揽高端人才，审慎研究、推进产业并构

金融危机爆发后，有些企业裁员并停止了招聘。对我们来说，这给我们在优化并完善人力资源体系上一次难得的机遇。我们已经开始着手吸纳国内外优秀的人才加盟。同时，危机也为本土IT企业在海内外进行产业并购带来机遇，浪潮也在积极的寻找适合的机会。

我们要正确认识和应对危机。“危”的意思是危险，“机”的意思是机会。除了以上几方面，国家为了保持增长、拉动内需，进行了4万亿的投入。这将带来政府以及相关企业信息化投资规模的增加，也为我们的业务发展提供了机会。我相信在危机到来的时候，只要做到反应最快、态度最务实、方式最合理、最齐心协力，我们的企业就会从这场危机中获得成长、取得进步。



浪潮集团董事长兼首席执行官 孙丕恕

成长型企业如何成功？

最近10年来，中国的大型企业发展很快，成绩斐然；同时，成长型企业也开始兴起，犹如雨后春笋，这里有很多成功的案例，但潜在的管理问题也逐渐暴露出来，急需解决。

管理涵盖的内容很多，我认为对于成长型企业来说，首先要把握的是一些最基本的、最简单的管理理念和方法，简单的道理不只是容易理解，同时也是最管用的，是很多目前管理问题的根源。

如何用我们中国人的文化智慧来解读西方管理理念？首先我们来探讨管理是什么？“管”是管人和“理”是理事，就是这样简单的道理！而做管理者的职责又是什么？“职”是指岗位，即在这个位置上要做的是什么；“责”是指对结果负责，要达到什么效果。管理者的职责就是“策划”（Planning）和“执行”（Execution），“策”是策略，“划”是计划，“执”是执掌、执控，“行”是指行为。我希望大家从中可以看到汉字是一种意义非常深广的文字，甚至可以解读现代的管理。

我把“管理”、“职责”、“计划”和“执行”这四个关键管理名词用中国人的智慧来分拆、排列，即形成对管理的一个全面观念的认识，同时也带出了我认为是做事的好方法：Plan-Do-Check-Act。

做管理其实很简单，因为到头来就是一个人把一件事情做好；难的是有众多人在不同的时间和地点，分工合作地做不同的事；换言之，管理的难点在执行。与大型企业相比，在管理上，成长型企业更应该把精力放在执行管理上。大型企业一般已形成了一套管理的体系和习惯，它的重点是要定出适宜的策略和周全的计划。成长型企业的情况有些不同，它们的策略大都是被动的，计划也相对比较简单，但因为业务快速发展、新人员大量增加，而管理体系还不成熟，所以执行力管理成为决定其企业成败的关键。

我在这里对成长型企业管理者提出西方管理的两个很基本的管理理念：

一是管理的基本原则：通过众人的力量来完成整体的工作，令每一个人把该做的事情做好。

二是戴明博士的Plan-Do-Check-Act（PDCA）。它原本是戴明博士在八十年代提出来用于生产质量管理的一种管理理论，被称为戴明环。该理论曾对质量管理实践起到很大作用，尤其是对日本的生产工业。我小的时候（六十年代），日本产品在国际市场上的名声是廉价、抄袭和低质，那时的日本产品质量就像现在中国成长型企业的管理。戴明的质量管理理论被日本接受后，经过几十年来的发展，现在，日本货已经被公认为是质量过硬的产品。简单的PDCA就有这样的力量！

其实做任何事情都可以运用PDCA，这个道理太简单了！做事情先要有一个计划（策划），工作的时候要掌握重要信息、控制关键环节（执行），作经常性的检讨（检查），从经验中学习、改进（修正）。将PDCA运用于企业层面，会形成企业的管理框架；运用于分公司层面，会形成团队共识的管理体系；运用于业务部门层面（例如销售、项目工程），会成为日常工作的执行管理模式和规律；运用于小组/个人层面，就是行之有效的工作习惯和方法。

现在，有些成长型企业使用了管理软件，如ERP，甚至CRM，对流程进行全面严格的管理，每个细节都不放过。但我认为，在企业的管理还没有到相当水平时，这些软件系统可能会成为负累和问题。成长型企业并不是没有类似的管理软件就不能运行，只是事事要老板亲自管和理；人和事多了，有时就管不过来，管理很难成体系，自然成功就无从谈起。所以我认为成长型企业如果把执行管理做好、把PDCA这样的好工作方法落实，获得成功就不会像想象中的那么难。■

谢克人

新锐国际有限公司董事长 谢克人

也谈中国信息化长尾市场

长尾理论由美国人克里斯·安德森提出。该理论认为，由于成本和效率的因素，过去人们只能关注重要的人或重要的事，而忽略需要更多精力和成本才能关注到的大多数人或事。在网络时代，由于关注成本大大降低，厂商有可能以很低的成本关注“尾部”，而且关注“尾部”产生的总体效益甚至会超过“头部”。安德森认为，网络时代是关注“长尾”、发挥“长尾”效益的时代。

分析国内现状，我认为，中国信息化也存在长尾市场。经过用友华表的实际调研，国内政府、企业存在着以下4种信息化应用现状：

1. Excel：对于某些政府部门或中小型公司，他们的信息化工具常用的就是使用MS Office，其中Excel使用尤为普遍。

2. 财务+ Excel：调查中，大部分企业信息化就是用友U8 + Excel或财务通+ Excel。部分走访企业认为：这种现状的原因，一是内部流程梳理和规划困难，二是目前的管理软件不能随需而变。

3. 财务软件+某一关键应用+Excel：如果是销售型企业，或销售管理比较规范、客户分类比较清晰的企业，除了财务软件外，它们会使用CRM软件；但对于其他的业务管理还是使用Excel管理。

4. 全面ERP应用+ Excel。使用ERP管理企业的核心业务，但在核心、关键业务之外还有大量的边缘信息使用Excel来管理和协同。如：邮件+Excel模式等。

为什么企业信息化普遍用Excel处理？第一个原因就是微软Office的大规模普及。第二个原因是企业信息化的需求五花八门。从业务的重要性和市场空间来看，CRM应该比财务软件市场更大。但事实却不是这样。

北京用友华表软件技术有限公司总经理 苗峰

国内主要ERP厂商60%的收入主要来源于财务软件，这些厂商财务软件收入多的达10亿以上，少的几个亿。而国内CRM厂商市场占有率第一的企业，年收入才几千万。之所以造成这种状况，就是因为财务管理比较规范，厂商好做标准产品；而CRM需求不规范，所以做出来的产品只能满足一小部分客户的需要。

正如CRM一样，企业还存在着大量的业务需要信息化。就是因为需求不一样，所以这些信息化领域的解决方法很难形成产品。因此，对于这部分需求，如果需要信息化，只能定制开发。定制开发有两种模式，一是自主开发；二是外包开发。

虽然Excel是世界上最伟大的软件产品之一，但它毕竟只是一个电子表格软件。要使用Excel进行信息化管理，因数据是文件方式存储，没有数据库，所以很多东西实现不了。其次是数据无法共享，难以对Excel数据统计分析。第三，Excel作为一个文件，分布在各个计算机中，数据很难汇总统计。最后也是最重要的一点，Excel数据都是非结构化的，对大量Excel数据很难进行统计分析。

在企业信息化长尾的头部，有财务、ERP、OA等软件，国内大多数管理软件厂商都在做这方面业务。而在信息化的尾部，市场上存在着大量的需求。目前这些需求都在使用Excel来替代，如何抓住这部分长尾市场，将是我们未来几年的使命！

如何与开源社区共发展

开源运动已经在国内开展了很多年，许多国内企业都在基于开源软件做自己的产品，可是鲜有几个能真正为社区做贡献，给国外同行的感觉是中国的开源企业只索取不奉献。究其原因，无非是两点：一是策略不明，不把开源作为公司的基本方向之一；二是不知道如何切入开源社区，进入不了核心领域。

2008年11月，由北京红旗中文贰仟软件技术有限公司（简称红旗2000）和北京大学承办的第6届OpenOffice.org世界开源大会在北京成功举办。为期三天的大会吸引了来自30个国家和地区的600余名国际开源专家参会，成为历届专业开源会议中规模最大的一届。

这次北京之所以能获得此次世界性开源大会的举办权，与红旗2000技术团队两年来所做的贡献和公司的开源策略是分不开的。正是在这些策略的指引下，红旗2000的开源技术部已经对OpenOffice.org进行了中文本地化改进；积极开展对OpenOffice.org的bug修改；同时正在独自进行或合作开发OpenOffice.org子项目。

红旗2000是如何在两年内取得如此成绩的？在这里我想重点谈谈红旗2000的经验，特别是如何与开源社区相处的原则：

- 不和OpenOffice.org竞争的策略：RedOffice的技术方向将遵循OpenOffice.org的技术方向，在重大的技术问题上将不会采取违背OpenOffice.org发展方向的策略。
- 回馈OpenOffice.org社区的策略：RedOffice的发展得益于开源社区。我们将遵循LGPL原则，把自己的工作成果全部贡献给社区，并建立开源社区。
- 努力影响OpenOffice.org的技术方向：也许我们无法左右Openoffice.org的技术方向，但我们要积极参与开源社区，增强我们在开源社区的发言权，使这些需求尽可能成为社区主导的方向。
- 不囿于开源社区的原则：我们不能被OpenOffice.org所束缚，需要根据产品设计，采取灵活的技术手段。这两年，我们在掌握了核心技术的基础之上，开发了具有独创意义的全新界面，并初步得到用户的认可。
- 来自社区回报社区：红旗2000严格按照LGPL要求回馈社区，并通过自己的开源，促进OpenOffice.org的发展。通过OpenOffice.org的发展，给大众提供满足需求、符合消费水平的多元化产品。
- 融入社区，项目引领，开放共享，重点突破：融入社区是第一步，在此基础上不能仅修改bug，还要敢于承担项目，这样才能逐步掌握核心技术。在这个过程中需要采取开放共享的方式方法，和国际专家进行合作。另外，需要选择自己熟悉的领域，有重点有步骤地深入进去。

正是在这些原则指引下，我和我的团队风雨同舟、并肩作战，终于在国际社区打开了局面。另一方面，基于开源、融入社区后，红旗2000技术团队的实力也随之增长。我们重新设计了全新的界面架构，让OpenOffice.org从框架上支持我们的新界面。这些事情在过去是不可想象的。这也是红旗2000一直坚持开源策略的原因所在。虽然说我们在开展开源合作上取得了一定的成绩，但要全面达到国际业界的水平，前面的路还很长。正所谓“路漫漫其修远兮，吾将上下而求索”，仅以此与国内开源同行共勉。

成修治

红旗2000开源技术部总监 成修治

本月焦点 2008盘点

10大美国IT业热门事件

美国的《ChannelWeb》杂志网站评出了2008年美国IT业最热门的10大事件。由于《ChannelWeb》专注于IT产品渠道分销事务报道，因此其中上榜的数项事件与美国IT产品渠道商有关。他们分别是：超薄笔记本大行其道、企业用户使用社交网站、“绿色IT”理念普及开来、智能手机层出不穷、微软汲取Vista市场失败教训、科技公司高管职位大变化、科技公司加强政界联系、赛门铁克遭遇渠道信任危机、思科被判违背渠道合同、美国科技业受到金融危机打击。

10大热门IT技术

准确把握IT技术的未来趋势，将在很大程度上影响到我们明天的技术体验和技术效率。国外媒体CRN网站在年末评出了2008年IT市场十大热门技术，虚拟化、SaaS和Linux均榜上有名。这些技术包括：虚拟化、SaaS（软件即服务）、802.11n高速无线技术、统一通信、视频会议、纳米科技、数据重复删除技术（De-Duplication）、Linux、高质量显示技术、大型格式化打印技术。

10大用户网络应用

2008年是互联网发展的重要一年，很多小众网络应用开始走向大众。下面是读写网第三个年终系列排名：Twitter、Firefox、IntenseDebate、Hulu、Ning、Last.fm、Meebo、Mogulus、Qik、Cooliris。

10大IT市场离职CEO

在即将过去的2008年，IT市场动荡迭起，CEO变动频繁。日前，知名IT网站CRN评出了2008年IT市场10大离职CEO，这10位离职CEO分别是：Bob Huang（IT服务供应商Synnex CEO）、杨致远（雅虎CEO）、约翰•汤普森（赛门铁克CEO）、陆思博（阿尔卡特朗讯CEO）、斯科特•科莱恩斯（Juniper CEO）、鲁毅智（AMD CEO）、黛安妮•格林（VMware CEO）、Lou D'Ambrosio（Avaya CEO）、埃德加•麦斯瑞（3Com CEO）、孔翰宁（SPA CEO）。

10大最佳软件公司

12月11日据国外媒体报道，美国知名IT杂志《CIO Insight》根据企业价值、可靠性和忠诚度三大标准评出了2008年10大最佳软件公司，传统软件公司的排名普遍靠后。这10家公司分别是：谷歌、红帽、Citrix、Adobe、Novel、Salesforce、 微 软、Cognos、Business Objects（SAP）、甲骨文。

10大IT意外事件

美国知名IT杂志《CIO Insight》评选出了2008年IT市场10大意外事件，包括：科技无法阻止金融风暴、虚拟化时代来临、云计算、绿色IT、苹果走进企业市场、企业创新也许只是幌子、云计算是祸还是福、社交选举、互联网越来越强大、Windows并未衰落。

10大谷歌公布上升最快搜索

据国外媒体12月10日报道，随着2008年的即将结束，囊括了全球60%搜索业务的全球搜索巨头谷歌，对外发布了2008年网络上最受欢迎的搜索字词。其中全球上升最快搜索为：萨拉•帕林、北京2008、Facebook登录、Tuenti、希斯•莱杰、奥巴马、nasza klasa、wer kennt wen、2008欧元、乔纳斯兄弟。

会议

“2009 IT影响中国”将在北京举办

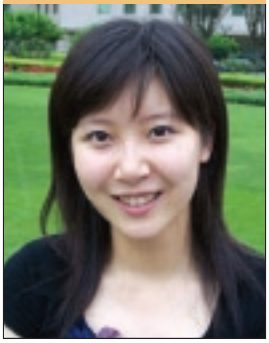
由中国计算机用户协会和中国互联网协会主办，天极传媒集团和中国互联网协会网络营销委员会承办的“感动•行动——2009 IT影响中国”于2009年1月6日在京举办。本次活动旨在通过对2008年度IT产业的重大事件、风云人物以及技术、产品等众多方面的调研、解析，探讨IT产业对中国社会经济产生的巨大影响以及未来的发展趋势。活动全程历时两个月，将覆盖北京、上海、广州、深圳、重庆、西安、成都、沈阳、武汉、厦门等城市，深入全国千万网民和40余座核心IT卖场，将“IT影响中国”打造成全民参与的IT盛宴。

2009中国IT产品创新与技术趋势大会

2009年1月9日，2009中国IT产品创新与技术趋势大会将在北京新世纪日航饭店正式举行。本次大会汲取往届IT168技术卓越奖颁奖大会与中国IT技术精英年会之精华，顺应开放式创新之潮流，除强化开放式评选年度技术卓越奖产品外，还将在海选2009技术趋势的基础上，举办若干热点技术发展趋势论坛，纵论2009技术大势。

中国互联网品牌大会

由中国互联网协会支持指导、人民网与国务院新闻办主办的《网络传播》杂志联合发起的“中国互联网品牌大会”，即将于2009年1月9～11日在北京举行。据悉，本次大会除了探讨网站品牌经营，交流经验外，还将发布中国互联网行业各类网站的十大品牌，发布中国互联网站品牌发展报告，发布中国互联网站品牌评价体系等。



陈阳
GNOME.Asia
网站主编

GNOME 2.26新功能展望

最新版本的GNOME 2.26将会在2009年的3月18日推出，在最新版本中将会有很多功能和结构上的改进。

从用户的角度来看，外观方面，将会引入一些新的桌面主题，如针对小屏幕的简洁控件主题；GNOME电源管理中ConsoleKit和PolicyKit将会更好地集成，以进行权限管理和安全性控制；媒体方面，Pulse Audio Mixer将会替代以前的GNOME音量控制器；在加密方面，在Pidgin和Telepathy中将使用SeaHorse作为新的加密插件，在evince中也可以用SeaHorse进行数字签名；同时在Nautilus文件管理器中将会新增工具栏编辑器，交互式的文件列表视图和文件关联管理。

从开发者的角度来看，GNOME 2.26将会有很多结构上的重大变化。DeviceKit将会逐渐取代HAL接口，因为其具有结构简单、扩展方便和易于维护的特点。所以在GNOME 2.26版本中，GNOME Power Manager将会首先使用DeviceKit来和硬件打交道。另外一个比较大的变化就是在GNOME Accessibility中将使用D-Bus来取代原来的CORBA协议，D-Bus具有轻便和兼容性好的优点，它不仅占用磁盘空间小（只有几十K大小），而且可以在KDE中使用，这有利于GNOME在移动设备上的应用和GNOME Accessibility在整个Linux世界的推广。

焦昕秋
CSDN
执行总编

什么样的云是安全的

安全是与每个人相关的事情，所以对时下安全界曝光率极高的“云安全”概念进行关注和理解是必要的，即使“云”这个词已经开始让你多少有点“审美疲劳”。

简单概括一下，现阶段“云安全”产品主要分成两大类。一类是基于互联网数据库的轻客户端程序，另一类是通过网状的大量客户端进行异常行为监测，将截获的信息推送到服务器端进行自动分析和处理，然后再把解决方案分发到每个客户端。请注意这两种产品中“云”的区别，第一类中服务器端是“云”，第二类则是客户端组成了“云”。不过，二者都有引人担忧的地方。比如服务器“云”模式，对外来威胁可以进行组合、判断、拦截，但一旦有病毒或威胁通过其他渠道入侵到用户的计算机当中，“云”是否还能对本机的安全威胁进行有效感知？再比如客户端“云”，能感知用户计算机上已经存在的未知病毒，但其是否具有拦截未知病毒入侵计算机的能力呢？如此想来，这两种云似乎更应该结合在一起。

“云”是不是在炒作？哪种模式更好？这事其实基本跟用户“无关”，因为他们需要的只是最终的安全效果而已。云多点没关系，用户也没什么可怕的，怕只怕光说不练，纯玩概念。

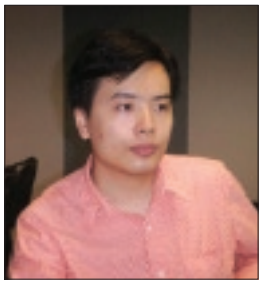
韩锷春
甲骨文大中华区开发者计划高级经理

Web应用快车

基于Web的数据驱动型的应用已经非常普遍，这些应用可能是使用Java EE或.Net技术构建。但是，在日常工作中我们仍然经常会用电子表格或个人桌面数据库去处理数据的收集、汇总和报表等工作。使用电子表格或个人桌面数据库固然很方便，但同时也会带来数据录入格式不统一、数据汇总工作量大、没有方便的Web访问界面和数据安全性得不到保证等问题。而对于这些临时性的、简单的需求，求助于Java EE技术、.Net技术或其他脚本开发语言似乎又有点“杀鸡用牛刀”的感觉。

Oracle Application Express（Oracle APEX）是一个用于Oracle数据库的快速Web应用程序开发工具。仅凭借一个Web浏览器和有限的编程经验，就能开发和部署快速、安全、外观非常专业的应用程序。Oracle APEX既结合了个人数据库的质量，又具有企业数据库的生产效率、易用性和灵活性，以及Web的安全性、集成性、可伸缩性和可用性。Oracle APEX是一个构建基于Web的应用程序的工具，应用程序开发环境也基于Web，非常方便。无须学习脚本语言或复杂的部署框架，只需编写几个查询并从已经构建好的用户界面主题和表单控件中选择即可创建高度专业、安全且可伸缩的应用程序。

微软技术



■ 主持人：Kaneboy
涂曙光，微软（中国）有限公司产品技术专家，博客堂成员。专注于.NET开发，从事Office System、SharePoint等产品相关的技术支持。

微软技术领域2009年快速展望

作为2009年度的第一篇文章，笔者希望能和大家一起展望微软技术领域中将要发生的“大事”。

.NET Framework 4.0作为整个微软开发技术领域最核心、最底层的技术，.NET Framework 4.0将成为一个更成熟的开发平台。其中新增的并行运算、进程内多版本CLR支持、新的PIA模型等等特性，无一不是开发人员非常关心或期待已久的功能。

PDC 08大会上，微软正式宣布了云计算平台Azure Services Platform。利用Windows Azure，开发人员可以构建从云端到企业数据中心，跨越PC、互联网和手机的各种应用。它包含了Windows Azure、Live Services、SQL Services、.NET Services等多组子服务。笔者期待在2009年，Azure Services能进一步的“落地”，使得所有的.NET开发人员，都能利用到云计算的能力。

现在的Silverlight已经将目标定位到了专业的开发人员以及企业身上。除了包含更丰富的功能之外，Silverlight 2已经完整地提供了对.NET的支持，你可以使用任意的.NET编程语言来创建Silverlight应用程序。对于企业，微软也在Silverlight 2中提供了与商业应用相关的控件，来协助企业构建包含富媒体功能的应用。

Windows操作系统每一次在界面、API上的变化，都会衍生出无数新的第三方应用程序。Windows 7是下一版本的Windows桌面操作系统，它将利用强大的软硬件优势，帮助消除信息、人和设备之间存在的障碍。Windows 7中新的界面元素、对触摸屏操作的良好支持，以及新增的API，都会给开发人员带来更多的机会。

Office 2007中全新的UI界面无疑是一个大胆但成功的决定。Office 14将进一步提高Office客户端与SharePoint 14进行交互的能力。而SharePoint 14除了将增加Office Web Applications，允许用户通过浏览器，直接在Web上查看和编辑文档之外，也向着企业级Web应用平台的目标继续前进。■

Java技术

■ 主持人：范凯
网络ID Robbin，JavaEye社区的创始人，开源软件的积极推动者和倡导人。

JavaFX重磅推出

Java社区12月的头条新闻无疑是JavaFX脚本语言的推出。JavaFX是Sun公司开发的基于Java的小脚本语言，语法和Java类似，可以直接调用Java的API；而且语法简洁，无须编译。写好的源代码可以直接在Java平台上运行，JavaFX带有强大的桌面组件库，用JavaFX编写桌面小应用，比Swing容易得多，开发效率也高很多。

JavaFX是Sun推出的新一代RIA技术，用于快速开发桌面应用和Applet应用。在JavaFX发布同时，Sun也推出了NetBeans的JavaFX插件，所以你可以直接在NetBeans里开发JavaFX桌面应用。

JavaFX除了可以运行在桌面以外，还可以运行在移动设备上；除了支持桌面组件，还可以支持媒体播放、音频和视频、动画等。因此JavaFX也被看成是Adobe Flex、Microsoft Silverlight的竞争对手。尽管JavaFX的发布有些姗姗来迟，但是以Java语言的全球市场占有率来说，也未可小视。另外由于Java占主流的企业应用对RIA方面的需求也越来越多，因此JavaFX有望填补Java在RIA领域的空白。

除JavaFX之外，Sun动作连连，NetBeans 6.5版本正式发布。NetBeans近年来颇有超越Eclipse的声势，而6.5版本的NetBeans在很多方面超越了前面的6.1版本：1. NetBeans的运行速度有了巨大提升，6.5版本的NetBeans速度飞快，已经比Eclipse当前的3.4版本快了很多；2. NetBeans支持了所有主流的Java应用服务器，提供了良好的Hibernate/Spring/Struts 2.0、EJB3、JPA等框架的支持，此外还对J2ME提供了完善的开发环境；3. NetBeans提供了对Groovy/Grails、C/C++、Ruby、Python、PHP的支持，集成了强大的JavaScript/CSS的开发工具，难怪NetBeans可以狂妄地喊出：“The only IDE you need！”

本月JBoss社区也有大的动作，前呼万唤始出来的JBoss 5.0正式版本终于发布了。JBoss 5.0可谓Java应用服务器的集大成者，它完善的支持EJB 3.0，支持Java EE 5.0，支持JBoss Seam 2.1，支持Hibernate3等。■

软件工程·项目管理

■ 主持人：潘加宇

UMLChina 首席专家，潜心研究和实践 UML/UP 相关技术的应用。

期待新一年

在过去的一年中，软件工程除了迎来它的40周年以外，总体来说还是比较平淡的。2008年发生的一件大事是IBM完成了对Telelogic的并购，使它成为Rational部门的一部分。在2008年12月，IBM发布了基于Jazz的需求定义工具—— IBM Rational Requirements Composer，它的仓储可以在现有的Rational RequisitePro和Telelogic DOORS需求管理工具之间共享。IBM宣扬Requirements Composer是“省纸”的工具，它将改变项目中产生大量需求文档和表格的局面。

Requirements Composer意味着IBM弥补了原来缺少的、开发生命周期中位于RequisitePro之前的工具板块，杀入需求定义工具市场，和市场中的Ravenflow等厂商在一定程度上形成了竞争关系。Ravenflow宣称自己的产品“为IBM Rational量身定做”（当然，也支持Telelogic DOORS），推崇Ravenflow RAVEN→Rational RequisitePro→Rational Software Modeler的需求建模路线。

RAVEN的亮点是需求的可视化验证，它可以分析英文文本，产生可视化的模型，以此和非技术的涉众交流，其中的Specification Checker功能还可以发现常见的遗漏需求。另一家公司iRise也宣布它的产品iRise Connect为Rational Requirements Composer提供类似的可视化功能。

另一件大事是微软重新加入了OMG。不过，微软并不采纳OMG的元对象设施（MOF）规范，而是开发自己的建模语言，称之为M语言。现任微软VSTS 2008 Architect版本首席架构师的Steve Cook认为这没有关系，因为M语言可以和UML相互交换，“MOF的核心价值就在于它定义了XMI（XML元数据交换），我们的工具将支持XMI”。

UML规格新版本2.3的修订目前正在进行之中，将于2009年上半年完成。UML就像数学里的公式符号和音乐里的五线谱。如果软件开发方法学没有大的变革，UML版本保持稳定是正常的。经常会有人关心自己“用的是UML 1.x还是UML 2.x”，其实都差不多。■

Open Source

■ 主持人：叶亮

系统分析员，SCJP，Java 工程师，关注开源社区。

金融危机下的开源世界

RedHat和Google签署了团体贡献协议，正式采用GWT作为JBoss中间件核心组成部分。从GWT 1.0 RC1版本开始，RedHat将用其作为redhat.com在线存储、客户中心和产品激活应用的一部分；同时，Drools/BRMS UI和Seam 2.1都已经使用或集成了GWT。RedHat认为在AJAX应用领域不需要另外的Java框架，不过承诺继续为Rich Faces和ajax4jsf提供赞助。

Google Chrome开源了，但之前只有Google自己的开发人员可以提交代码，外部人员只能提供补丁。现在，Google允许外部开发者成为Chrome的正式开发人员。根据文档中描述的指导政策，成为Chrome的开发人员需要担负一定的责任，协助项目达到目标，并且必须曾贡献过10到20个重要补丁，还得获得三个以上不同“人物”的同意。同时，Google Pack中推荐的浏览器已经由Firefox改为Chrome。

说到Firefox，Mozilla将很快终结Firefox 2的开发。他们的目标是将注意力转移到Firefox 3.1和移动版Firefox的开发。在过几天，Firefox 2的用户将收到升级为Firefox 3的推送通知。随着Firefox 3.1正式版的发布，Firefox 2将逐渐消失。

正当全球金融危机愈演愈烈的时候，不少企业重新审视中国市场，纷纷来找机会。Sun董事长McNealy在Sun Tech Day期间访华，向国家民政部减灾中心捐献一套Black Box，并且重申坚持开源的决心。随后，MySQL 5.1正式发布；并且在北京召开MySQL 2008用户大会。接着，Java平台的新成员、用于创建RIA应用的JavaFX 1.0版正式发布。

另外据传，中国移动联合Google、联想推出Ophone，即基于中国移动开放式手机操作系统OMS、定制中国移动特色服务的联想Ophone手机。同样是据传，中国移动研发的自有手机操作系统OMS（Open Mobile System）已经获得相当进展。但是“据传”何时能成真呢？中国企业的确需要在移动终端方面多一些话语权。■

移动开发技术

■ 主持人：马宁
微软最有价值专家，Windows Mobile 开发者。

移动元年列传

年之前的专栏里，我曾经有一个预言：移动世界的战国时代到来了。回顾过去的2008年，移动世界的格局变化是巨大的。

iPhone的优势似乎仍然无人可及。但是当用户尝鲜的热情减退后，就会发现iPhone是用户友好的代表作，但从来不对开发者友好。仓促推出的MobileMe并不成功，这开始让人们对iPhone神话产生了一丝怀疑。

本年度的最佳新人是Android，在经历了SDK风波后，Android样机终于如期上市了。比起老牌的手机操作系统，Android还显得十分稚嫩。不过Android在技术方面的领先性是无可置疑的。但是随着功能的扩展，Android的开发团队是否能够保持稳定的质量将是一个巨大的挑战。

山寨机在今年升级为一种文化，而且在不同领域被广泛传颂。这个看似强大的巨人，却被死机短信点中了死穴。MTK的潜能已经被压榨干净，MTK的应用程序错误可以让整个操作系统崩溃。所以山寨机在操作系统上的升级换代已经势在必行，现在看起来Android最有希望替代Nucleus，不过这要依赖MTK方案的升级。

在应用程序上，跨平台似乎成为方向。Gmail和Google Map提供了Windows Mobile版本，微软发布了针对iPhone的Seadragon软件，而Adobe向Windows Mobile授权了Flash Lite。其实主题只有一个：当操作系统的功能逐渐弱化，软件厂商开始提供跨平台的软件或者服务。

应用程序在线商店是移动应用的热点，最热的是iPhone的App Store。微软和Google也在很短的时间内发布了自己的在线商店。但是我认为Google与其在App Store上投入精力，不如将自己的广告平台尽快移植到手机应用中来。

最后一个热点是浏览器，Windows Mobile停止了DeepFish的服务，转而发布了IE Mobile 6.0。iPhone和Android的浏览器都是基于Qt的Webkit，基于Firefox的Skyfire和老牌手机浏览器Opera也推出了新版本。

浏览器模式真的适合移动互联网吗？2009年，也许会给我们答案。■

数据库技术

■ 主持人：王翔
软件架构师，主要研究方向为XML、.NET、领域设计和PKI应用。工作之余喜爱旅游、写作和烹饪。

Oracle发布EDA

Oracle新闻站点近年来有关其Oracle数据库的报道比例似乎在萎缩，除了经常看到某家知名企业选择Oracle数据库的报道外，大部分有关产品发布的报道都是关于Oracle自己企业软件的内容。

本月，Oracle发布了新版事件驱动的Fusion中间套件产品——Oracle EDA（Event-Driven Architect）Suite，该产品是从BEA Weblogic Event Server嫁接的产品，主要通过基于Java的OSGI技术便于企业在发现、过滤、分析及相关性判定的基础上，把各种事件（更直观地说就是商机）呈现给用户。相对于耗时但不一定真正“切题”的商务智能而言，Oracle EDA力图提供更实时的商务决策支持。类似其他Fusion套件，EDA主要包括下列内容：分布式/本地时间缓冲、与Oracle Coherence分布式/本地缓冲服务的集成、事件的记录和重放功能、可视化管理Console以及仅针对Java开发的扩展工具。从使用上看，Oracle更希望将数据从管理对象，变成激励业务的火花。套件安排上Oracle将JDeveloper、Business Activity Monitor、Business Rule、ESB以及Complex Event Processing安排在一起，务求将数据与业务规则，以及触发的事件连在一起，让用户可以在更严峻的商业环境中，把控“现在”的商业机会。

另外，在试图追赶竞争对手IBM数年之后，本月Oracle发布了新的主数据管理产品——Oracle Site Hub MDM。区别于其他MDM产品，该产品的特色在于能够把企业分散在多个商业网点以及不同物理位置的信息进行集中的MDM管理，通过它将相关的库存、资产、不动产信息与商业网点结合在一起，此外借助集成Google Map，使得该产品特别适合各种层次型IT部署的企业，例如：具有多层次及多家分支机构的零售行业、公共服务领域、具有多种前端资金交互要求（比如：具有分支银行、ATM机和POS机）的金融服务性机构。

区别于以往针对个别领域“零星”的Fusion中间件的发布，这些产品综合体现了Oracle近几年在数据库管理、企业级安全管控方面的成果。■

Web 开发

■ 主持人：钱宏武

职脉网技术合作人，原搜狐互动产品开发部主管，资深互联网社区架构师，垂直搜索领域专家。

2008 年终总结

对于现在按照西方规则运行之下的商业和互联网，12 月一般是一个开心的月份，而今年年底西方带给我们的却是经济危机。今年的终结大概所有人都会很小心，否则就真的完全自由了。

看过无数裁员的新闻，我们估计慢慢也会习惯。而这次晚节不保的是杨致远。作为互联网前辈标杆式的人物，黯然离开的结果，让我们这些后辈们诧异无比。真是应了那个“我猜到了开头，没猜到结尾”的感叹。坊间分析文章无数，老人家也上了本年度最不合格 CEO 的宝座，人脑袋上帽子无数，估计这个帽子是他最不希望看到的。

同样老牌的 W3C 推出了新的 Web Accessibility 标准。我一向对于推标准的组织很是敬重，认为这些人才是互联网的灵魂。这次提的标准，目的是用于帮助开发者设计利于残障人士使用的网站和在线内容。我一向对于所谓的 Web 2.0 比较迷惑，W3C 这次提出的 WCAG 2.0 才是真的 2.0。 更多 Web 开发的新特性、新标准，这些是构成了互联网前进的真正动力。

同样提供基础服务的 Sun，在这个冬天特别的活跃，一方面凑云计算的热闹，推出了自己的云计算服务部门；另一方面，收购的 MySQL 在本月也推出了 5.1 版本，从我最初使用 MySQL 3.X 开始，到现在已经是 5.1 了。作为 MySQL 的忠实 FANS，很欣喜地看到这个发布：更快的速度，更多的企业特性，更好安全设置。MySQL 是开源界一个优秀的数据库，Sun 则是一个不卖软件卖服务的先行者。

我一向认为是做操作系统内行、做互联网门外行的微软竟然也推出了一个开源的 CMS，基于 SQL Server。开源是开源了，水不贵，可装水的桶太贵了。

今年的 SD2 大会是很出乎我意外的，这么萧条的经济形式和这么火暴的场面形成鲜明的对比，看来“科学技术是第一生产力”永远不会过时。

2008 年就要过去了，回想起来，今年对我们来说，确实是一个最好的年代和最坏的年代，一个天堂与地狱并存的年代。■

游戏开发

■ 主持人：赵青

九艺科技 CEO，曾任金山西山居技术总监，网易方舟产品总监，《剑侠情缘 2》项目负责人及主程序员，资深游戏开发者。

美女与野兽

以前，我这个做研发的一直搞不懂“性感美女”和游戏有什么关系，为什么游戏的广告中那么多和内容不相干的暴露，为什么游戏展会上的 showgirl 们越穿越少，后来想明白了这个简单的道理：游戏的用户大部分是男性，并且是年轻的男性，他们喜欢这些，所以产商就用这些来吸引他们。当然，广告也有优劣之分，有些做的有品位，有些做的很隐晦、很色情。

碰巧最近看到了一个罗列网游行业雷人广告的网站，摘录一下，大致有这样几类，一类是色情擦边球，为了不毒害年轻的程序员们，这里不再举例。另一类是误导，欺骗型，比如“永久免费”，“来玩就送 600 元”等，还有那种都运营了很久，还在说“今天震撼公测”的。

也许有人会问，为什么网游行业挣了那么多钱，还在做这些不入流的事情？我想有几个方面，一是这个行业时间还比较短，暴利吸引了大量的人员、资金进入，难免有些歪瓜劣枣，时间久了他们自然会被淘汰。二是用户的素质也需要提高，我一直认为，在这个趋利的时代，有什么样的用户就有什么样的厂商。在厂商的实力没有达到很高的程度前，引导用户是一个空谈，适应用户是必然的选择。

私服、外挂、打钱公司一直像一只野兽，危害着网游的正常运营。尤其对于那些运营情况比较好的游戏，由于利益的引导，这些野兽有着充足的动力和技术能力，寄生在这些游戏周围。从技术能力上，外挂的制作者是一些熟悉系统底层，精通逆向工程，甚至有一些游戏开发经验的程序员们。他们更加贴近用户，知道用户的需求。通过长期的斗争，大家都认识到，仅凭技术能力，只能改善，不能根除他们。如果想更有效的解决这类问题，就需要法律和强力的执行来配合。

韩国可以给我们一些借鉴，目前等待韩国国会审批的游戏法修改案上规定，对外挂制作和销售人判处 1 年以下有期徒刑和一千万韩元（折合人民币约为 4.65 万元）以下的罚款。并且，NCSOFT 公司等 10 多家游戏公司也正在准备用法律武器对付出售外挂的公司。■

安全

■ 主持人：肖新光
安天实验室首席技术架构师，研究方向为反病毒和计算机犯罪取证等。

辩证法之 12 月

12月微软IE7的一个0day漏洞不期而至，利用方法被迅速传播，恶意页面泛滥开来。

微软花了10天才发了一个补丁，而且根据补丁说明，微软“没有信心”完全修补这个补丁。这显然比MS08-067被动得多。

在这个漏洞中，我们发现了一些辩证而有趣的现象。很多安全研究报告建议用户采用对所有进程启动DEP保护的方式来回避这个漏洞，但其结果让人理解了为什么微软过去不敢以此作为默认策略。那就是IE、Office以及大量采用纯C库的软件都会因此出现不稳定的情况，而且甚至不能添加到DEP的例外中去。要安全还是要稳定？成为一个真实的选择。

另外一个有趣的现象是，对这个漏洞防护表现最突出的，是一个叫锐甲的免费小工具。这个只有2M的工具可以监控IE的异常行为从而拦截木马。但这个小团队也花了一天的时间，修补DEP完全打开带来的不稳定。看来安全工程师斗争的对象不仅是入侵也有微软的“神经代码”。事实上，Vista的PatchGuard、UAC等机制曾经带给我们很强的信心。DEP甚至让一些漏洞挖掘者十分悲观。但当微软也变得“没有信心”时，选择第三方安全软件还是等待微软的补丁也变得十分富有辩证味道。

本月，Wi-Fi手机的开放之争又打到了公众面前，很多人把禁止Wi-Fi完全算到WAPI标准的账上。只认为这是WAPI惹得祸，或许并不公平。如果运营商本身坚决，像移动、联通这样的企业对决策的影响能力也是很大的。可以简单试想，如果Wi-Fi高度普及，那么更多人可能就会用Skype之类的语音通信工具来节省高昂的移动话费。

但从作为安全工作者，看到的则是另外的问题，那就是百万台走私手机带来的安全隐患。这里真正担心的，不仅是对个人隐私和安全的威胁，还有类似设备的无孔不入所能达成的渗透效果，以及有可能组成的“手机僵尸网络”的威胁。从安全策略的角度也可以看到一种有趣的辩证法。■

商务管理解决方案

■ 主持人：贾菡
解决方案专家，曾获微软Dynamics CRM产品认证专家、微软CRM MVP称号。有多年企业信息规划管理、企业商务管理解决方案的项目经验。

年末风云大作

IT业的又一个冬天不似今年的雪来得那么迟，而是早早地开始侵蚀软件厂商和实施公司的每个角落。乱世从来都是座次的重新洗牌，也是英雄辈出的时代。企业减少在IT方面的投入，对需要长期系统规划的管理类系统颇为不利；但由此提高的工作和沟通的效率，又为能提升绩效监督、优化流程和协作的管理类系统提供了最好的机遇。因此管理类系统的厂商现在将会更多地考虑调整产品设计、功能特点、性能优化，以及目标定位。

上月中旬，微软负责Dynamics产品线的商务解决方案部副总裁Kirill在丹麦举行的Convergence 2008 Copenhagen大会上发布了Dynamics NAV 2009。NAV是微软ERP应用套件之一，相比另一个应用套件Dynamics AX，NAV更注重并适合在中小企业中应用。此版本的最大亮点之一是增加了“Personalized Role Centers” 工具和数据访问功能，它是为21种工作职能量身定制的。这是微软一直希望在其商业解决方案系列产品中包含并加强的，力图达到让不同用户角色访问其相应功能模块的效果，使之更专注自身的工作，而其余的功能都放在幕后，由此提高系统的安全性、操作效率。新版本同SQL Server集成地增强了商务智能方面的表现，让数据挖掘和智能分析更好地融入日常工作中。

在1999年因为自主的SCM软件受到DELL和通用等顶级客户追捧，i2科技公司市值一度飙升至284亿美元，而如今却被其竞争对手JDA软件公司以3.46亿美元凄惨收购。在收购了数家SCM厂商之后，JDA开始觊觎SAP和Oracle在其领域的霸主地位。但是在年中，其授权收入大幅缩水，有人认为这是因为它并未对收购的软件做更好改进和融合。相反同样有重量级客户，并开展了一系列收购的Red Prarie和Ariba，因为有自己的专注领域并不断优化改善，都有不错的业绩。

2008年末注定风云大作，孰起孰落，还需等待。引用前人一句美文：“繁花三千，落下的瞬间，你看见了谁。”■

动态语言

■ 主持人：高昂

资源与环境信息系统国家重点实验室博士生，关注动态语言，喜爱写旅行游记，同时也是OSGeo中国和InfoQ中文站成员。

谁让动态语言归来

Python 3000 最终版本已在12月发布，新的Python 3.0与2.x版本相比，在字典和字符串等众多内置对象细节方面发生了较大变化，并且标准库的部分内容也已进行重新组织。

伴随新版本的发布，Python社区关注重点将会逐渐转向Python 3000上，对于已有代码，如果您打算向新的平台迁移，可以尝试官方SVN储存库中提供的代码自动转换工具Python 2to3。2to3工具可以根据参数将指定文件或目录代码转换成Python 3.0，当然，2to3工具需要运行在最新的Python 2.5环境下才可以顺利完成转换。

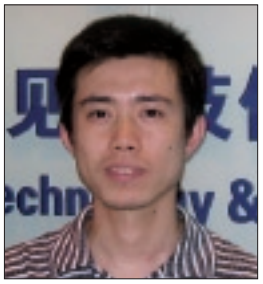
Python语言以其设计优雅和开发高效而著称，NetBeans IDE也为Python开发者推出仅有28M大小的NetBeans 6.5 Python Early Access版本，在IDE中提供了高亮显示、代码折叠、智能感知等代码编辑特性，并同时支持Jython和Python两种运行时环境。

昔日王者Smalltalk，最近由于其Web框架Seaside的日益火热而受到众多关注。这门早在20世纪七八十年代就已开始流行的语言，曾对Ruby的发展产生过深远影响，发明人Alan Kay也是图灵奖得主。

Seaside构建在Smalltalk实现的Squeak虚拟机之上，可以帮助Smalltalk开发者进行Web应用的快速搭建，目前Seaside最新版本是2.8，在这个版本中，提高了程序运行的性能，并且较之上一个版本的内存消耗降低到原来的四分之一。RoR之父DHH曾在自己博客中高度评价Seaside的优异表现。

毫无疑问，IT领域每年都会有众多新事物诞生，同时，已有技术也在新技术的带动促进下发生着不断的演进，一旦时机成熟，便会展现出新的活力。在一定程度上，这些无疑都为整个开发技术的繁荣做出贡献。从Python到Ruby，从Smailltalk到Erlang，还有Lisp、Perl、Groovy以及轻量级Java脚本语言BeanShell，不论是老枪或是新兵，越来越多的动态语言正在走进开发者们的视线。■

嵌入式开发&硬件



■ 主持人：孙天泽

华清远见嵌入式培训中心金牌讲师，嵌入式行业资深专家，畅销书作者。

新年·新品·新气象

2008年悄然逝去，可能用“不平凡”这个字眼都无法体现出2008年的与众不同。在这个同样是绕太阳公转一周的365天里，我们得到了比以往多得多的震撼：南方雪灾、汶川地震、北京奥运、神七升空、毒牛奶、金融风暴……酸甜苦辣各种滋味势必会烙印在每个人的记忆中。虽然金融危机蔓延全球，影响了整个世界的消费能力，但是这丝毫没有阻碍科技的不断进步和创新，在2008年的科技新闻中，不断有能够刺激我们眼球的新品出现。

先看一下美光科技有限公司推出的串行NAND闪存技术，该技术可以使嵌入式应用产品能够方便地升级存储容量。随着操作系统功能的不断增强，用户存储的多媒体数据量越来越大，GB量级的存储时代很快就会到来。美光串行NAND闪存的最低芯片密度为1GB，处理速度为2.64 MB/s，写入速度比NOR闪存更快。客户能够轻松地在串行NOR闪存目前的存储能力之外，以高性价比扩展存储容量。

随着2009年的到来，已有分析师对全球芯片市场的发展趋势进行预测，其中一条预测是英特尔的Atom处理器将被应用到上网本以外的更多设备中。对这一点我深信不疑。在当今的笔记本领域，Intel占据了绝大部分的市场，丝毫没有抵挡不住AMD等对手的意思。为了让Atom更好的深入人心，Intel更是推出了Moblin平台，从而简化开发难度。虽然有些忌惮微软、谷歌的反对，Intel在推Moblin时很低调，但是这并不能掩饰Moblin的技术优势。我们在关注Android的同时，不要忽视了Moblin的存在。估计2009年，会有更多的基于Atom+Moblin的产品上市。

在嵌入式领域，还不敢说Intel是ARM的最大竞争对手，但是两巨头的口水仗倒是不少。在Adobe 2008 MAX大会上，Adobe与ARM宣布了一项技术合作协议，两公司将Adobe Flash Player 10和Adobe AIR技术进行优化，使其能够应用于基于ARM的产品。这足以表明ARM对Intel的质疑是很在意的。通过iPhone的成功，厂商们已经开始注意到用户体验的重要性。ARM此举意义深远，优化过的Cortex-A系列处理器将成为现阶段最优秀的平台。■

动态语言

■ 主持人：胡长城
网名“银狐999”，普元软件R&D部门架构师。在Workflow、BPM、SOA领域有过多年的产品研发和实施经验。

SaaS与SOA

即将步入年关，对2008年的企业应用做点归纳性点评。我个人只想提两个方面：一个是SaaS的技术浪潮值得关注；另一个是SOA复合应用模式的落地。

随着2008年12月5日，阿里巴巴的叶伟、赵进等人编著的《互联网时代的软件革命——SaaS架构设计》一书在“软件开发2.0”大会上首发，可以说国内SaaS应用已经跨越“概念炒作阶段”，步入了“技术探索阶段”。未来几年里，除了商业模式上的突破和竞争，技术架构和实现的百家争鸣，也会非常精彩。这是一个SaaS技术浪潮的时代，传统软件厂商必须开始考虑将其架构和产品往可支持SaaS模式的架构上迁移或改造。

但国内SaaS应用依然还只是“技术探索阶段”，目前国内SaaS厂商基本上还都是“各自为政”。应用开发商提供的应用很难通用，迁移性和重用性都受限于SaaS平台提供商。解决这样的问题，估计只能等待某一天，国内这些SaaS平台提供商和大型SaaS应用提供商能够围坐在一起，围绕一套统一的标准、架构模型来构建平台或应用。

相比较“SaaS”，似乎下半年有关SOA的声音很单薄。在刚刚结束的“2008年中国软件技术大会”和“软件开发2.0大会”中也少见SOA的身影。其实，SOA已经悄无声息的由先前的“技术探索阶段”开始步入“标准化的应用开发阶段”了。

两件事情值得软件开发商非常关注，一件是Garnter提出的“SOA复合应用”的概念，另一件是明年初OASIS组织即将发布的SCA（服务组件架构）规范的1.1版本。前一件事情代表一种理念和趋势，而后一件事情则是实现这套理念的标准。目前主要中间件厂商都已经在产品中提供了对SCA规范和复合应用的支持，并基于此提供对复合应用的开发模式的支持。可以说，在未来几年里，应用开发商们在开发SOA应用的时候，只有一种架构和开发模式可供选择，那就是符合SCA标准的复合应用开发。■

RIA技术

■ 主持人：王保平（花名玉伯）
资深前端开发工程师，崇尚简洁而不简单，相信付出才有收获。就职于淘宝网UED部，忙并快乐着。

前端与民工

最近看到一个很让人沮丧的比喻：程序员是IT中的民工，而前端则是IT民工中的掏粪者。我们来分析一下究竟有没有道理。

可以把网页的开发比作成楼房的修建。修建楼房，首先要有建筑图纸，这是网页开发中的原型。有了图纸，就可以用钢筋水泥搭建楼房了，这些毛坯房是网页中的结构。毛坯房需要装修，这就是CSS。JS则可以看成是楼房里的电梯。没有电梯的楼房，有楼梯依旧可住，只是电梯会让住户更方便一些。在网页中，电梯和楼梯反映的就是渐进增强的开发思想。

好像没啥可讲的了。中国的毛坯房，都是民工来搭建的，砖瓦水泥等材料也没啥新花样。毛坯房的搭建，技术含量很低，民工们很可怜。作为前端，在大部分公司的地位极低，所用的砖瓦，无非也就是HTML的90多种元素，因此一类比，可以得到一个很明显的结论：前端就是民工。

太让人沮丧了！…… 等等，我们来分析点有意思的。

第一，我们的砖瓦比建筑民工多。我们有90多种元素可用，什么时候该用什么，建筑民工们照着规律来就行。但是作为一名前端开发工程师，如果不去思考，那就真的和民工无异。但只要一去思考，我们就有可能变成巨豪。《老人与海》的魔力究竟在哪儿？用词的精简和选词的恰当是海明威的魅力所在。因此，想要成为一个优秀的前端，也必须在选词和用词上花大工夫。

第二，我们的装修能力比民工强。作为前端，如何装修，这是前端自主选择的。比如布局方案，采用经典的float布局还是先进的双飞翼布局，这取决于对CSS知识的掌握程度和涉猎的广度，学习成本比民工高多了。

第三，民工会安装电梯，我们前端是能做电梯。JS固然威力巨大，但光有电梯是无法构成楼房的。如果没有良好的HTML代码，JS就是空中电梯，只能看看，真要坐上去，会摔得很惨。

铁肩担设计，妙手写网站。前端是一个目前还未被大众理解的但迟早将熠熠生辉的新生职业，我们不是民工！■

CSDN 网站、《程序员》杂志总裁蒋涛开场致辞

雪中送炭，王者归来

——2008 软件开发 2.0 技术大会后记

■ 记者 / 周至

一场能为软件开发行业带来巨大影响的业内技术盛会应该具备哪几个特点呢？业内精英齐聚？演讲内容高端？举办规模庞大？如果你有这样的疑问，那么由CSDN与《程序员》杂志社联合主办的第二届软件开发2.0技术大会就能够让你得到一个明确的答案。

此次大会自12月4日开幕，历时三天。主办方邀请到包括现代软件之父Ivar Jacobson，Amazon云计算战略师Jeff Barr，IBM首席架构师寇卫东，金山副董事长、UCWEB董事长雷军等国际和国内50多位知名技术专家。大会围绕新一代互联网、企业应用技术、软件工程管理、语言与工具、移动时代、软件创富等六个议题，开设了数十场主题演讲和课程，并与参会的千余名技术专家互动、交流，分享经验与心得。此次大会是本年度讲师阵容最强大、内容最丰富的技术饕餮之一。举办之时恰逢软件“寒冬”之际，风雪交加；而该会议的第二次举办恰似一团熊熊烈火，帮助大家度过寒冬。正所谓“雪中送炭，王者归来”。

CSDN总裁蒋涛在开幕致辞中表示：“金融海啸后的中国软件既有危机也存在契机。中国软件业目前面临着两个问题：一是整体环境相对恶劣，知识产权受到的重视不够，用户对软件价值的认可也需得到进一步提升；二是企业规模小、效益差，养不起好的人员。而由于缺人，所以技术也弱。归根到底一句话，持续发展能力相对较弱。但是云计算、SaaS、开源、开放平台这些软件业的趋势，都给中国的软件同仁提供了一个很好的开始。此外，中国是一个增长很快的国家，有很好的硬件基础。所以，只要我们能够做到天时、地利、人和，便会迎来一个中国软件行业的黄金时代，冬天也会变得美丽，中国软件产业也会有丰硕的收获。”

技术•天下•势

30年来，软件技术飞速发展，给IT产业、世界经济和人类生活带来了翻天覆地的变化。从PC的诞生，到图形用户界面的兴起，再到互联网的大行其道，每一次技术变革都引发产业重塑，影响一代企业和个人的兴衰荣辱。从面向过程，到面向对象，再到如今的面向组件、面向服务，软件思想的每一次升级，都带来软件生产方式的巨变，以及IT产业链条的再分工。每个IT人都知道，技术趋势的变化犹如天时，顺之者盛，逆之者衰，兹事体大，不可不察。

在这次SD2.0技术大会上，主办方邀请到了诸多来自国内外著名IT企业的专家们，以“技术•天下•势”为题，从各自不同角度与在场观众分享了他们对于时局及产业发展的精彩观点。

IBM软件集团两岸三地大中华区总架构师寇卫东在“思路决定出路，创新才有未来”演讲中指出了软件行业的五大趋势：软件行业将向超大规

模发展，并购之风盛行；软件企业之间的联盟正在形成，国内也形成了例如长风联盟之类的组织；软件向公开标准化发展；软件向整合、优化、简单化发展；软件长尾蓬勃发展。在展望未来时，寇卫东认为，纯粹依靠外包拉动增长的方式不可长远发展，中国软件将以企业应用为驱动，借本地化优势，基于开源与标准化，逐步在应用软件、通用软件、嵌入式软件等领域争取长足发展，进而在开发工具和平台软件等方面有所建树。

Oracle公司全球软件研发中心副总裁Frank Xiong，为与会者揭示了“互联网创新与企业级应用的新模式”。谈到企业运作中低效率高成本的瓶颈问题，他指出要利用网格的优势来解决。

现代软件开发之父Ivar Jacobson也为与会者带来了精彩的主题演讲“产业何处去？技术何处去？”。在演讲中，他讲述了敏捷开发对于未来软件开发的影响以及如何利用明智的开发方法为企业创造财富。

“站在巨人的肩膀上，才能看得更远”。同样，在这些业内专家的指引之下，众多软件开发者才能看清行业的发展趋势，把握行业变化的脉搏，顺势而动。

杳霭祥云起，且看云深处

在大会“新一代互联网”议题中，云计算无疑是最受关注的话题。大会主办方亦看到了这一趋势的颠覆性力量，而让其作为今年这一议题的主角，登台亮相。Amazon云计算战略师Jeff Barr在其“云计算，未来这几年”演讲中提道：“之所以云计算环境受到开发人员和创业者的喜爱，并为他们提供了更多机会，主要基于以下几个原因：云计算能够帮助开发者将精力集中在自己独特的技术和业务优势之上，使其在基础架构建设上花费的时间更少；使软件在可扩展性和成长性上遇到的问题更少；节省在服务器上的资金投入；使业务增长变得更加容易。”


众多国内外专家云集一堂

对此Jeff大胆地推测，十年之内，应用和服务以云计算的形式提供和使用，将是再平常不过的事情。人们自如地使用服务和应用，而根本不会关心它们来自于哪里，以及如何提供给我们。

谷歌中国开发技术推广部、中国地区首席项目经理栾跃也在演讲中介绍了谷歌针对云计算在推广开发技术上的三大理念和愿景：

1. 让云计算，即以网络为基础的计算功能、服务以及数据更易于获取。

2. 使得网络的链接更加普及和广泛应用。用户能够通过更多方式更多设备更容易地访问网络。

3. 让云计算客户端（浏览器）的功能更加强大。随着网络应用的不断扩大和深入，未来计算机应用的发展重点将不再仅仅是基于桌面计算机的单独应用软件，而是基于浏览器的应用。

此后，谷歌中国工程研究院副院长兼工程总监幺宝刚也以共同打建云计算时代的开放网络平台为题作了精彩的演讲。

会上的演讲以及会下的交流，无不使与会者对于云计算这一计算机技术发展变革中的重大趋势有了更清晰的认识。这其中包括：云计算这一趋势产生的必然性；该趋势所带来的技术变革以及硬件环境的变革；如何利用其盈利等。只有明确了这些问题，中国软件企业才能更早地把握这一先机，以备日后增强自身核心竞争力。

英雄背后的故事

在这次技术大会上，与会者不仅见到了平日里难得见到的技术“大牛”，也能够有机会了解这些软件“英雄”成功背后的故事。

鉴于很多开发者对首款iPhone中所涉及的技术充满兴趣，主办方在此次大会上特别邀请了首款iPhone中文输入法iCosta的作者之一李亮为大家带来了“iPhone平台应用开发和实例解析”的精彩演讲，讲述他在开发这款输入法时的亲身经历。

在谈到开发iCosta的初衷时。李亮回忆到，他的朋友去年九月份从美国带回来一部iPhone，当时他就被这款手机的绚丽应用吸引住了。唯一的遗憾是这款产品并没有中文输入的应用软件，于是他便致信给苹果的工作人员。但是邮件如石沉大海，没有任何回应。无奈之下，他便和朋友一起动手编写了这款中文输入软件iCosta。

Amazon云计算战略师Jeff Barr激情演讲


金山副董事长、UCWEB董事长雷军和与会者热烈交流

提及苹果在iPhone应用程序开发上的政策，李亮回答："出于自身商业模式的考虑，苹果开放的只是iPhone应用程序最少化的接口。应用程序开发后如果需要放到苹果的网上商店去，则需要通过苹果的验证。"之后，他又对iPhone SDK中采用的语言Objective-C进行了较为详尽的介绍。

此外，金山公司董事会副主席、UCWEB董事长雷军也从他的职业生涯经历谈起，为一些希望成为技术企业老板的技术人员提出了一些建议。他认为成功的技术企业的老板或者创业者，必须知道用户需要什么。所以，假如程序员想创业的话，就需要补充这样的知识，然后还需要找到一个可以迅速增长的机会。

与会者无论对于这些业内"英雄"的职业经历还是他们所提及的相关技术都表现出了极大的兴趣。演讲结束了很久，仍然有很多人围绕在他们身边讨论问题不愿散去。

九华论剑，谁主沉浮

晚间进行的主题沙龙讨论活动无疑是此次大会上"群雄"各抒己见，交流"切磋"的最佳时机：戈壁合伙人有限公司徐晨、曾担任IBM Rational Software总经理的Steve在"技术创业与风险投资"沙龙中与与会者探讨了技术创业和风险投资的问题。徐晨认为，只有项目创始人做到了以下几方面的准备，才有可能获得投资：丰厚的投资回报——有爆炸性的增长；对风险实际评估——有足够的风险防范意识；详细而务实的财务计划——3-7年的可行退出策略，上市和并购都是VC比较好的退出方式；独特的价值定位——即使看上去相同，也要有足够的差异化。

在"感受和思考调试器的威力"沙龙中，英特尔亚太研发中心高级软件工程师张银奎围绕"感受和思考调试器的威力"这一主题，通过生动的演示和讲解，就调试器的实际应用和高级使用技巧与与会者一起展开趣味的学习和生动的讨论活动。本次沙龙的主要内容包括：使用调试器快速定位应用程序和系统崩溃的原因；使用虚拟机调试BIOS和引导过程（除了使用开源的Bochs虚拟机外，还将使用英特尔的SoftDV工具——强大的全平台模拟器）；使用内核调试理解计算机系统的软硬件；使用ITP/XDP（基于协议的硬件工具）来调试纯软件调试器难以调试的问题；程序指针飞跃等高级的调试技巧。本次沙龙的最大亮点是调试器与病毒的攻防演示，与会者表示最感兴趣的内容是调试器的工具应用。此外，张银奎讲师为此次沙龙准备了数十本《软件调试》，这些书作为有奖提问的奖品都被争抢一空。

其他几个沙龙的会场里也都挤满了渴望着交流的与会者，有些人甚至担心错过其他沙龙的精彩演讲而穿行于各会场之间。直到夜色已深，很多人仍旧不愿离去，因为他们所需要交流和分享的内容还有太多太多。

小结

CSDN软件2.0大会在众多讲师的无私奉献、与会友人的热情支持下，秉持着最初的理念，积累经验、不断创新，飞速的发展、成长着，致力于通过自身的努力为大家呈现一幅中国和世界IT产业格局的完整画卷，帮助中国软件企业把握产业变化脉搏，顺势而动。我们有理由相信，软件开发2.0大会这个国内技术盛会的"王者"必会如海上明灯一般指引软件开发行业的前进方向！

全民动员，保证互联网安全

■ 记者 / 郑柯

相信你一定还记得：在四、五个月之前，为了保证那场世界体育盛事的顺利进行，北京城的大街小巷布满了穿着同样白色T恤的男女老少。看上去大家在随意谈天，神态放松，可心里却都绷着一根安全的弦。一旦发现任何异常情况，一个电话，就会让可疑分子无以遁形。“全民反恐”的形式，对于保证整个赛事的安全起到了重要作用。而瑞星公司于2008年11月18日在“2008年瑞星互联网安全技术大会”上公布的“云安全”策略，将这种形式搬到了日益凶险、危机四伏的互联网上，号召大家群防群治，全民防控，在互联网上打一场针对病毒、木马的人民战争！

可能有人会问：有必要吗？瑞星公司《2008年度中国大陆地区电脑病毒疫情&互联网安全报告》指出：2008年的病毒数量继续暴增，比2007年增长12倍以上，其中“网页挂马”所传播的木马、后门等病毒占据90%以上。从木马病毒的编写、传播到出售，整个病毒产业链已经完全互联网化，这是导致病毒数量暴增和危害增大的根本原因，获取经济利益是病毒作者的根本目的。

病毒产业的互联网化，使得黑客可以利用互联网来加速病毒的制造，提高制造病毒的效率。黑客采用的三大手段包括：

● 采用效率更高的病毒生产软件，这些软件可以通过加壳、加花等方式，把已有病毒改造成杀毒软件无法识别的版本，从而可以自动生产出大量新木马病毒。

● 租用更好的服务器、更大的带宽，为“木马下载器”下载病毒提供硬件上的便利。由于黑客产业的丰厚利润，黑客团伙有经济条件改善自己的“生产环境”，以求更丰厚的利润。

● 利用互联网论坛、博客等，雇用“软件民工”来编写更强的驱动，并将其加入木马中与杀毒软件对抗。

面对整个黑客产业大大提高的病毒制造效率，传统的反病毒软件和厂商面临如下挑战：

● 新病毒巨量增加、单个病毒的生存期缩短，现有病毒监测技术无法及时截获新样本。

● 即使能够截获，每天高达数十万的新样本数量，也在严重考验着反病毒厂商对于病毒的分析、处理能力。

● 即使能够分析处理，如何能够让用户在最短时间内获取相应的病毒库，也成为一个重要的问题。

传统的反病毒思维，都是基于被动的防御和反应模式。而瑞星本次提出的“云安全”计划，是指安全的互联网化，是将用户和瑞星技术平台通过互联网紧密相连，组成一个庞大的木马/恶意软件监测、查杀网络。这样每个用户都为“云安全”作出贡献，同时分享其他所有用户的安全成果。而“云安全”网络则充分利用了智能行为判断技术，将云端计算机的事件及时提交到云安全中心响应。在云安全中心，威胁信息分析以及来源挖掘将会同步进行，快速响应用户的威胁，阻断威胁源。

在接受《程序员》记者专访时，瑞星研发副总经理刘刚指出：在3—5年之内，计算机安全行业必然是要被互联网化的；应对危险时，不再是单向，而是双向的，会大量运用类似云计算、分布式计算、并行式计算这一类技术，构建整个网络的防御，而不是单点防御。单点防御只不过是整体网络防御的一个基础。当被问及有关用户隐私的话题时，刘刚说：“云安全”不是无度地提取客户端的所有信息，而是只提取跟“威胁”相关的、智能分析后的信息，并且这个提取发送到服务端的行为是完全可以被用户控制的。用户可以自主选择加入“云安全”或不加入“云安全”，不加入也还是可以享受安全保护。当然，参与者越多，整个网络就越安全。■

寒冬中的远见

——达内科技暖冬工程

■ 记者 / 鲁兵

金融危机在今天已经成为人们见面必谈的重要话题之一了。这在IT领域也不例外，自雅虎大规模裁员之后，其他不少IT公司也开始出现类似情况，微软、IBM、Google等平时极为重视人才的企业，此时的招聘也几乎完全断绝，投资商们更是捂紧腰包，留住有限的现金流好过冬。在这样的背景下，达内仍然获得了大笔的现金投资，并计划启动暖冬工程，不得不让关注软件的人们侧目。

11月25日，达内科技“暖冬工程”的发布会显得暖融融的。这不仅仅是因为酒店的空调效果不错，更重要的原因是达内科技这次宣布的项目，对于整个IT行业的影响都是暖洋洋的。

达内科技CEO韩少云在大会上介绍了“暖冬工程”的具体实施方法：“三年内投资1亿元现金，用作垫付参加培训学员的学费。学员在找到工作后的12个月内将学费返还给达内。”整个工程运营体系影响到的是四大类群体：高等院校大学生、达内公司、达内培训合作伙伴（如微软、Sun等）以及国内软件企业。

如果回头看看高等教育领域的情况，我们就能感觉到就业形势的严峻。2009年多达610万的高校毕业生正面临着前所未有的就业挑战。这当中有相当一部分学生是计算机相关专业。尽管“不务正业”对于中国大学毕业生是司空见惯的情况，但这样庞大的就业队伍，对于任何行业都是一个烫手的山芋。

而2009年，几乎已经被认定是整个业界乃至整个经济系统受金融危机影响最严重的一年。原来那些对人才最重视的企业，此刻也已经关闭了招聘的大门。即使是在名牌高校，也渐渐有不少大学生开始为毕业之后是否能找到工作而担忧。继续学习，似乎成了不少学生暂时逃避困难的出路。这些人当中，有些直接参与社会的就业培训，这在IT领域尤其如此。

然而，数以万计的学费价格不菲。已经完成大学教育并给家庭带来负担的大学生不在少数。再次拿出这笔费用来完成就业教育，对很多人来说都是一个巨大的门槛，这也是“暖冬工程”出现之前很多大学生以及就业培训企业遇到的严重问题。

2005年12月，达内首先推出了“零首付抵押金”的培训模式，用韩少云的话说：“这在中国的教育和培训行业里面是非常创新的培训模式。”而事实上，达内的数据也告诉我们，通过这种模式培养出来的大学生，能够很快找到工作，并还清培训公司的借款。这与前面“暖冬工程”提到所影响的最后一类企业有很大的关系：它们得到了需要的人才，这也为整个系统运行提供了保障。

正是因为敢于在收费模式这种关键问题上进行创新，达内获得了IDG的第二轮投资。在这个寒冷的冬天，这笔投资的分量比以往任何时候都要显得沉重。韩少云表示，达内将会在未来的三年时间中，分别投入4000万和两个3000万，完成这一亿元的投资，让更多计算机相关的毕业生能得到培训的机会，从而给整个产业的冬天带来一丝暖意。

最佳实践经验助力软件企业发展

——微软“软件开发‘模式与实践’论坛”首次登陆中国

■ 记者 / 周至

2008年12月2日，微软“软件开发‘模式与实践’论坛”恰如一位阅历丰富的老者自北美来到有“六朝古都”美誉的中国古城南京，为中国的程序开发人员、高级用户界面设计人员和解决方案系统架构师等目标人群，提供如何将软件工程的方法论、软件系统架构设计模式及流程和微软的.NET开发平台进行集成的指导，并且提供微软官方的“模式和实践”展示。

此次活动由微软(中国)有限公司和南京软件园联合主办，主办方邀请到了在软件业享有盛誉的现代软件工程之父 Ivar Jacobson博士和微软公司架构策略总监John deVadoss等软件大师在会上和与会者分享了软件工程方法论和软件架构设计、软件程序开发、软件流程控制等领域的最新研究成果，及其在微软公司内部的最佳实践经验。这其中包括了他们历时9年所研发出来的工具、指导以及一些成熟的解决方案等。

在微软的开发领域中，像开发WINDOWS、OFFICE、SQLServer、MSN等产品时微软都在使用自己的Visual Studio产品（目前这个版本是2008，接下来会发布下一个版本2010）。而在开发这些产品的过程中所产生的超过三亿七千一百万行源代码，也在用微软Visual Studio Team System (VSTS)以及Team Foundation Server (TFS)系统套件来进行管理工作。总而言之，微软将其模式与实践的经验在他们开发的整个三亿七千一百万行代码中融会贯通，最大限度的提升了自身的开发效率。所以，我们有理由相信，这些实践经验完全可以彻底改进一个企业的开发方式，极大程度地提高他们的生产效率。

当问到这些实践经验改变一个企业的开发方式、提高该企业的开发效率需要多长时间时，John deVadoss指出：“这需要一段很长的时间，不是几个月就可以完成的。我们需要建立一些试点，在实践中总结经验，慢慢地改变企业原有的开发方式。”这一观点引起了Ivar Jacobson博士的回应：“改变一个人的行为很容易，但改变一个人的思想很难，平台和工具都可以在某一角度上提升企业的工作效率。但是，如果要彻底改变一个企业已经习惯了的开发方式，可能至少需要18个月左右的时间。另外，保持一种持续专注的精神是很重要的。”

谈及“在这些实践经验中，关于软件的规划实施方式应该是怎样的，自上而下和自下而上哪一种方式更好”这个问题时，Ivar Jacobson博士语出惊人：“当今，自上而下的SOA正在慢慢死去。”接下来，他举了一个例子详细地说明了这一点：“比如，我们开发一个软件产品，但产品的需求是在不断变化的，而采用自上而下的方式，会让我们在进行了一段时间的开发之后，突然发现之前规划的一些文档成了‘纸件’，即一些无用的东西。所以，我们需要设计的是一个可执行的架构方案。”关于软件开发流程的发展方向，Ivar Jacobson博士还提到：“大的流程也正在死去！”他认为，在现在的软件开发环境中，一个大流程想解决一切问题的想法已经不切实际。应该是用一个个经过实证的实践，由下而上的堆砌出适合这个开发团队的流程，并且与开发工具及开发平台密切的整合，才能够真正落实并产生效益。他认为，在这方面的发展，微软的Visual Studio Team System团队开发平台与Microsoft Solutions Framework for Agile Development流程方法处于世界的领先地位。John deVadoss表示非常赞同：“自上而下的方式不适用于现实的开发。其实，还有一个折中的办法，具体到SOA，就是‘middle out SOA’的概念，我们已经看到有客户通过这种方式获得成功，创造了商业价值。具体来讲，就是从中间开始，而逐渐向上、下两个方向延伸。”

目前，中国的经济发展受全球经济危机的冲击相对较小，但我们也应该时刻关注外面的局势变化。由于经济环境不太乐观，越来越多的公司不管是在中国还是在其他国家，在服务外包方面都会要求在降低成本的同时提高生产效率。简单地说，微软“软件开发‘模式与实践’论坛”所带来的最佳实践经验，能够在一定程度上帮助中国软件企业抓住这次危机中的契机，提高自身的核心竞争力，取得发展而迈上一个新的台阶。■

洞察危机中之机遇，把握自身命运

——中创软件公司高层主管张嘉平、陈致平专访

■ 记者 / 孟岩　　■ 文 / 周至

如今金融海啸席卷全球，而软件行业也难以独善其身。所以，软件行业何去何从自然成了当今的热点话题。在这个特殊的时刻，难道我们只能悲观的等待这个“寒冬”的过去？我们是否应该尝试去寻找这危机之中的契机？那么具体应该怎么去做呢？为此，记者采访了中创软件公司副总经理张嘉平先生，请他为我们解答这一系列的问题，并洞察这次危机之中的机遇所在。

张嘉平，原台湾三商计算机股份有限公司董事并担任执行副总，现任中创软件公司副总经理。

陈致平，曾担任IBM Rational Software总经理，负责大中华区的业务。现任中创软件公司副总经理。

危机之中，存在契机

谈到这次金融危机，张嘉平首先对市场环境进行了一个全面的分析。他认为，大陆有一个最大的特点，就是拥有巨大的市场，这也是70年代造就的亚洲四小龙所不具备的。当时，亚洲四小龙由于没有大的内需市场，所以做的东西都是面向外面的。

“我们的模式与当时的亚洲四小龙不同，我们要对自己的市场进行整合，让这个市场成为最大的收入来源。有了足够大的市场，就不会造成行业内的恶性竞争。实际上，我们的市场已经大到足以把国外开发出来的先进的东西，成型的、好的技术直接引进使用。”张嘉平表示。

除此之外，具有本身环境特色的软件还有更大的发展空间。因为这个根本不是国外的软件企业所能够开发的。现在的情况是，软件的自主创新品牌刚刚起步萌芽，所以可以暂且引进国外的先进技术并行使用。

他举了一个例子来说明这个问题：“CDMA不是我们发明的，但是最大的手机市场在我们这里，并且我们也拥有着中兴和华为两大世界知名企业。所以，我们一定要抓紧自己最具有核心竞争力的地方，并可以融合世界上已经成型的标准和组织。另外，我们应该建设具有自己特色的环境，来把这些东西融合在一起，创造一个巨大的市场。这样，我们就可以以逸待劳，事半功倍，因为我们有一个巨大的市场来支持。接下来，我们要做的就是首先在属于我们自己的市场中创造效益，而在国外的市场自然也就会形成了。大家都知道，美国现在所有的知名品牌都是先打开美国本土市场的。”

关于金融危机，张嘉平谈道，在去年年底，他已经从台湾的经济环境中看到了一些端倪。第一，比如台湾的高科技，从去年第四季度已经开始下滑了；第二，石油价格的飞涨，也是泡沫经济的一个前兆。总而言之，不管是不正当的泡沫经济也好，还是高科技的下滑也好，他发现台湾很多的企业都已经开始抱着现金，紧缩而不投资了，他确实已经感到有“警讯”传出来。

接下来，张嘉平对这次金融危机问题进行了较为具体的分析。他认为，不管是哪一次冲击，都会有同样的一个特性，就是会造成强者越强，弱者越弱，甚至消失。因为大陆的市场没有在虚拟的金融环境中做太多的琢磨，而是以实体经济为主，所以在这次冲击中，实体经济相对虚拟经济所受的冲击较轻。在这之后，大陆市场可能面对的就是强者越强，弱者越弱的情况。一些经得起冲击的厂商都会在市

场中寻找机会扩大，因此这些厂商都会到大陆来。

“当然，我们也接触过一些国外的厂商，他们之前都没来过大陆，但现在感觉他们希望能在大陆市场找到一些机会。所以，我们虽然会被金融危机波及，但我们要做到的应该是沉稳一点，抓紧我们的核心竞争力，并且把这一竞争力转化为扩大市场的利机。就机遇来讲，我们如果自己做不好，水也流不过来，所以我们应更加努力。这样当今的局势就会倾向于我们，世界各国的人都会乐意与我们做生意。当然，这些的前提就是我们首先要自己准备好。”张嘉平表示。

正如张嘉平所言，我们的市场前景广阔，并且存在着很多潜在的机遇，但这都需要我们做好准备，不断地发掘机遇，不断地迎接挑战。那么，我们具体需要怎么做呢？

完备自身体系，迎接机遇到来

希望在大陆开创一番新的事业，中创软件在软件管理方面的承诺等几个原因，让张嘉平在离开台湾三商集团之后选择了大陆的本土企业——中创软件公司。他本着这样一个想法：希望能够帮助中创软件在软工、体系方面打造得更好，让中创软件的辉煌不止1个17年，甚至是5个、10个17年，让它有序的经营下去。他认为，一个软件公司，只有体系稳固了，才能走得长远、走得大。

“一个企业要发展得好，一定是大、中、小环境都要好，大家都朝同一个方向努力，自然水到渠成。”张嘉平对这句话进行了特别的强调。中创软件一直致力于打造一个好的环境，首先把自己做好。这其中包括了对员工职业生涯的规划；令员工以开放的态度来看待中创软件提供的机会；坚持承诺在软件工程管理上的持续性；让整个创新的动能持续在企业里面产生；保证企业里面软硬件资产的安全等。并且，他认为，企业不应该只是把技术的环境做好，更多的是应该把工作的环境做到位，将公司大的环境中管理的、经营的每一个环节都做好，这样才能打造出一个可持续的体系，并借助很多项目来实践精进它。

“过去台湾石油危机时，台湾决定下一阶段致力于发展高科技的电子业。于是，所有的法律、高科技、进出口业、税务、教育体系、贸易人才全部规划来打造这个体系——尤其是对于市场人才的培训。那时候最突出的是经济区和国贸区，国贸区训练外语人才有他非常独到的方法。我认为只有这种完备的体系，才能创造出真正的核心竞争力。”张嘉平举了这样的一个例子，他认为，对一个软件企业也是一样，只有打造了一个完备的体系，才能帮助企业建立它的核心竞争力，在这次“危机”之中寻找到契机所在。

程序员何去何从

现在外界环境对IT界的冲击主要是在从业人员的心态上。尤其是近年来，很多大公司在招聘中均有缩水，而且部分出现了裁员的现象。这都导致了诸多技术人员人心惶惶，在职人员担心失业，而毕业生也为就业问题而忧心忡忡。当问到能否为中国现在的程序员提出一些建议，中国的程序员应该何去何从时，张嘉平回答：“我们的市场环境大都没有规划出一条清晰的、使程序员成长的路线。我不敢站在大环境上来讲，但从我们中创软件企业的内部环境来讲，首先，我们一定会规划一个蓝图给程序员看：你可以从一个最初级的程序员按部就班地变为一个‘老师傅’；可以将来做管理层的工作；可以走向更深入的架构设计道路；也可以变成好的讲师或顾问。”

所以，他认为，在这个特殊的时刻，程序员切记不要惶恐，而且每个企业都需要为新员工进行这样的规划。另外，现在世界上有这么多新生的技术，而程序员是真正站在接触技术信息最前沿的位置。所以，张嘉平希望程序员能够把这些信息进行有效的规划和整理，然后带入到企业体内，让企业体更加快速、顺畅地接收到信息。这样一来，不仅对企业有好处，更重要的是，体现了一个优秀程序员的价值。

关于应届毕业生的问题，由于陈致平前一阵子正在做“校园招聘”这方面的工作，所以对此深有感触。他从两个方面给这些年轻人提出了一些建议：对有机会被招聘到企业的人来讲，他希望大家要渐渐达成一个共识，帮助企业来完成它的稳定性，而不要经常的跳槽、更换工作，否则企业和程序员自身都是受害者。因为，程序员在不断更换工作的过程中，学习的过程也不稳定，会在一定程度上影响自身的职业道路发展。另一方面，没有找到工作的人也不要气馁。因为，市场现在不是处于等待的位置，而是处在一个开创的位置，这里面存在着很多隐性的商机。更何况现在有了互联网，有很多新种的业务都需要软件来增加它的附加价值。所以，如果这一部分人真的希望在互联网上创造一个新种的业务，找到了属于自己的市场并且有所把握，不妨去尝试一下。因为等待不一定是一个最好的策略，完全可以试着去奋斗，去创业。也许，数十年后属于互联网行业的佼佼者就是在这个时机下被创造出来的。

“我希望所有人都能很正面地去看待这个时机，我们拥有一个最大的且正在扩大、成长的市场，现在最不应该做的就是对它有一个悲观的认识。”陈致平表示。

对此，张嘉平也十分赞同：“总的来说，程序员要把稳定的地方做好，而企业对于程序员的生涯规划也要有很好的布置。两者结合之后，稳定了，良性互动了，整个软件行业也就稳定了。”

春节是平的

幽默

发传真

用户："我刚买了一台你们的计算机。你们不是说可以用计算机发传真吗？我已经试了快一个小时了，可还是发不出去？"

工程师："你的modem接好了吗？打开了吗？"

用户："模—代—木？什么是模代木？噢，是不是那个调制解调器？那我接好了，也打开了。"

工程师："那么，你能用它拨电话吗？ modem响吗？"

用户："响！我还能听到传真机接通后的吱吱声呢！"

工程师："那你的文件准备好了吗？"

用户："当然准备好了。"

工程师："那你按send键了吗？"

用户："按了。这些不都写在使用手册上了吗？我完全是按照说明书上的操作步骤一步一步做的。"

工程师："那就应该能发传真了，不应该再有问题呀！嗯……，啊，是不是你的文件格式不对？"

用户："文件格式不对？文件不就是写在纸上的吗？"

工程师："纸上的？你再说说你的操作步骤？"

用户："我先把我要传的文件放在屏幕的前面，然后用调制解调器拨号，听到接通的声音后，再按send键。另外，我觉着用计算机发传真挺费劲的，文件把整个屏幕都挡住了，我得站起来从上面才能看到屏幕上的东西，……"

工程师："……"

你要辞职了

一位网络公司的CEO把公关经理召来，说："我们的股票在纳斯达克上市不久就跌了下来，你想个办法让我们的股票在短时期内涨起来。"第二天，该公司的股票上涨了三个百分点，第三天又上涨了九个百分点。CEO非常高兴问公关经理道："你用了什么办法让我们公司的股票涨了起来？""我放了个假消息。""什么假消息？""我说你快要辞职了。"

电脑诊断

门诊部，医生对一位七旬老头说："不用怀疑，这是科学，只要将你的病情输入电脑，电脑就会给出百分之百正确的诊断！"老头："我头晕、恶心、呕吐、浑身乏力，还……"医生："噢，等一下，电脑说，你怀孕了！"

招聘员工的理由

某电脑经销公司经理来到人才交流中心，工作人员问他想招聘什么样的人才，经理说："希望能像CPU一样勤奋工作，最好还能超频；像鼠标一样机灵多智；像键盘那样一触即发；对待客户要像显示器一样面面俱到；对待工作像打印机一样一丝不苟；对待公司老板像主板一样兢兢业业。""那么他的薪水呢？""最好能像电脑那样不知疲倦不计报酬。"

繁殖老鼠

一所中学的电脑教室由于要使用Windows的关系必须要购买一百只鼠标（mouse），于是向财务部打了报告。不久就接到会计室的公文："因为经费有限，请先购买一对老鼠，以便繁殖后代。"

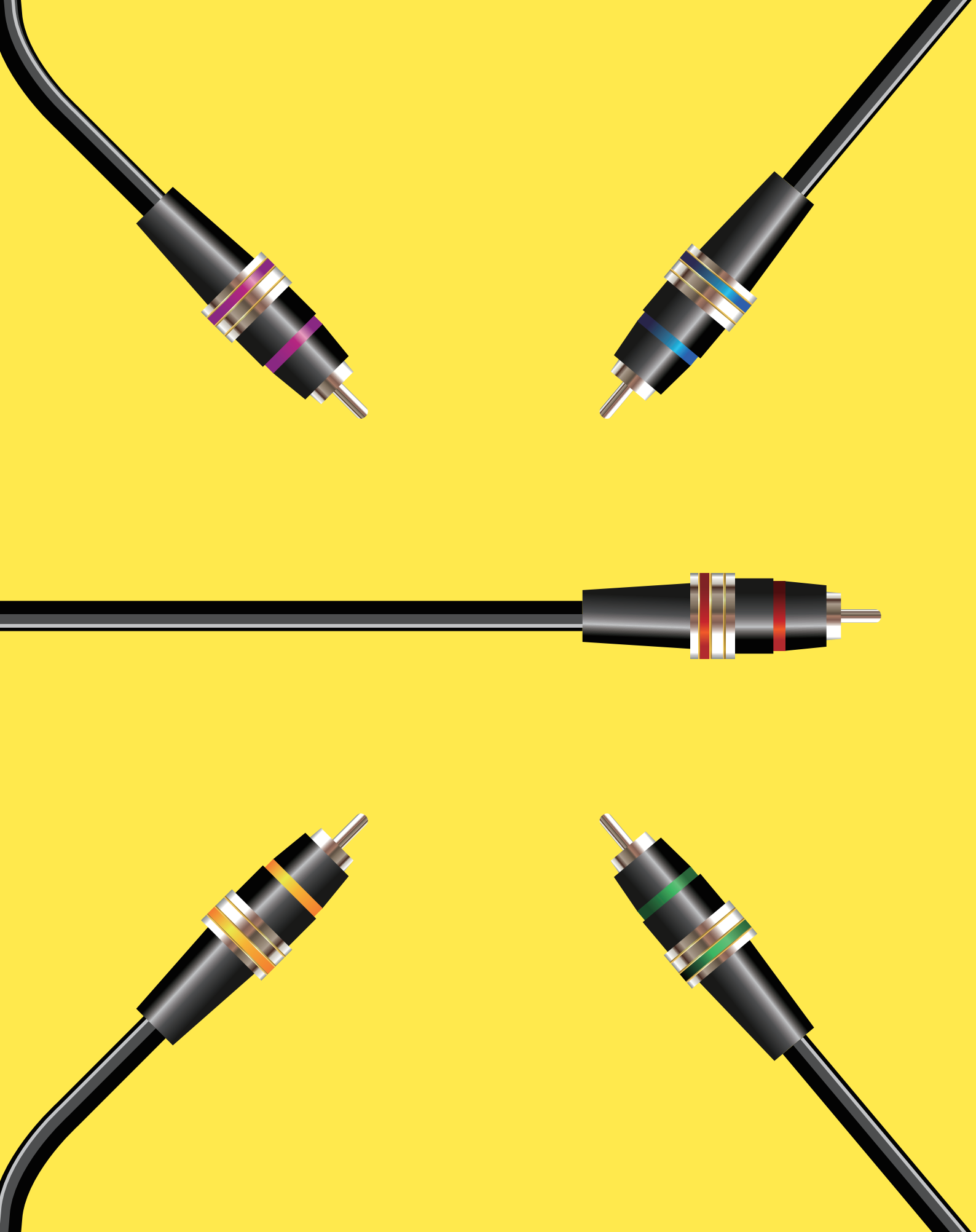
声音

中国软件现在比较弱，有了"天时"、"地利"，"人和"是下一步应该挑战和努力的方向。现在已经解决好了"地利"，"天时"也对大家是有利的，把"人和"做上去，中国软件的黄金时代就会来临，冬天也会有美丽的冬天，中国软件产业也会收获硕果累实。

——CSDN总裁蒋涛在第二届SD2C大会上的致辞

纯粹依靠外包拉动增长的方式不可长远发展，中国软件将以企业应用为驱动，借本地化优势，基于开源与标准化，逐步在应用软件、通用软件、嵌入式软件等领域争取长足发展，进而在开发工具和平台软件等方面有所建树。

——IBM软件集团两岸三地大中华区总架构师寇卫东在为SD2C大会上发表《思路决定出路，创新才有未来》时的精彩演讲



产品经理的5种能力

■ 文 / 欧阳璟

最近的一两年，我总是可以在各种各样的邮件列表以及招聘论坛中看到产品经理招募的信息。除此之外，还有很多自己创业的朋友也常常问起我是否有产品经理的人选推荐。印象里，CSDN的产品经理概念是从Blog产品开始的，当时的产品经理，对CSDN Blog有不小的推动作用。结合这个产品，很多项目逐渐推广开来，也让CSDN的Blog用户在很短一段时间内就超过了10万。后来，我开始回头考虑，产品经理的角色到底是什么？他们要想方设法得到企业高层的信赖与支持，他们要推动产品开发的进度，他们要用各种关系和资源推广自己的产品，他们甚至于要去完成很多细节的设计工作。

为什么忽然之间市场对产品经理的需求有了如此之大的跨越？

这就像在前不久CSDN主办的软件开发2.0大会上，CSDN掌门人蒋涛所提到的：Business of Software已经成为我们不得不重视的一项工作，这也是我们为什么要从《程序员》的角度来看产品经理的原因。

产品的价值

任何一项单独的技术，在它没有被产品化之前，其应用价值都可以被看成是0。这绝非危言耸听！在技术社区，我们常常可以看到很多对纯技术顶礼膜拜的开发者。这其中不少圣徒对于如何做软件、如何做市场、如何销售技术的工作视而不见，这也是不少技术创业失败的根本原因。

下面我们作一个假设：如果一项好的技术其应用价值为1，那么基于这项技术所开发出的软件产品应用价值可以达到10；当然，软件市场上产品很多，但好的产品在其中的比例不会超过10%，由于产品并不仅仅包含技术，那么我们可以认为一个做得好的产品的应用价值超过100；即使你完成了一个很好的软件产品，如果没有有效的途径销售，或者获取其他资源，那么它并不能成为一个商品，而成为商品之后的产品影响力再大一个数量级是可以预计的，这样一个商品所带来的应用价值也许就是1000了；更进一步，市场上流行的商品或者是领域内领先的商品，其价值再乘以十倍，其应用价值则达到10000！或许有些人会认为这种计算方法有失偏颇，但相信大多数人此刻都已经明白，软件产品的价值其实并不仅仅只取决于技术。

我试图去说明的是软件产品的价值点，它不仅仅在于技术，但技术在软件产品中同样非常重要。倘若没有最初的1，那么也绝不会有后来的10000。

当然，上面这个笼统的假设并不能全面概括技术、产品设计、产品开发以及推广等相关工作的真正价值。不同的产品，对应的不同市场，是应该有不同价值点的。以现在很多互联网产品为例，推广的模式和资源成为了限制互联网产品发展的一个重要瓶颈。

更重要的是，作为产品所在的组织，其组织架构以及工作流程等方面也面临着重要的挑战。仅在软件开发阶段，需求工程、项目管理、开发方法、质量监督等多方面环节都可以说具备非常重要的价值，这也是完整产品不可或缺的重要组成部分。所有的这些工作汇总起来之后，产品的开发便成了一件极为琐碎的工作。协调和平衡各种各样的关系、充分调配资源、准确的把握产品发展策略等工作，在原有的组织结构里越来越难以开展。

正因如此，产品经理的职位才显得格外重要。这也是为什么我们看到最近一两年，寻求软件产品经理人才的企业越来越多的原因。若是按照前

面假设的价值分布，他们正肩负着将技术价值从1放大到10000的压力，而他们所带来的价值，也是前所未有的。

产品经理改变世界

如果要给产品经理做一个严格的定义，相信每一个企业所给出的答案都是截然不同的。但毋庸置疑的是，几乎每一个成功的产品背后都有一个，或者一群人在担任产品经理职责。他们承担着产品成败的责任，有时甚至关系到这个企业的生存，这对于唯技术论者无疑是一个极大的讽刺。如果你看过市场上一些并没有太多技术含量的产品，就可以很容易地理解这一点。

那么对一个成功的产品，甚至是商品，哪些环节才是最重要的？在商品社会，生产和制造仅仅只是整个价值链当中的一小部分。经济学家郎咸平对于产业价值链环节的定义有个非常著名的6+1理论，其中的6包括：产品设计、原料采购、物流运输、订单处理、批发经营、终端零售，而那个1才是加工制造。

如果把软件产品当成市场上其他实体商品的话，那么6+1的理论同样适用于软件市场。技术所解决的问题，在很大程度上解决的是加工制造的问题，而真正在价值链上游的那个6，才是真正能够创造财富的环节。

产品经理所做的工作，就是解决这个6的问题。

对微软创始人比尔•盖茨的评价很多，但是如果要我们选择比尔•盖茨众多丰功伟绩中最重要的一个时，那么多数人都会认为将软件商品化是他对数字时代最大的贡献。不管你是桌面电脑终端的用户，还是一个大型软件公司的总裁，如果没有软件的商品化，没有软件授权的协议以及这种模式的广泛流行，也许今天我们生活的时代还很保守。不管自由软件、开源软件如何攻歼微软的产品授权体系，但你不得不承认的是这套体系主宰了我们20多年来的整个软件商品市场。而微软，借助整个价值体系的6，成就了伟大的软件霸业。

毫不夸张地说，作为产品经理的比尔•盖茨改变了这个世界。然而，像他这样的产品经理，毕竟在人类历史上也寥寥可数；对于一般开发人员来说，除了踏踏实实做好软件，认认真真写好代码之外，产品经理这一职业发展通道还需要我们具备更多能力与基本素质。

成为产品经理

通过对超过20名各种软件产品经理的沟通，我们对产品经理的基本素质进行了简单的总结，多数人赞同产品经理需要具备下面的5项基本能力：

- 规划力
- 设计力
- 沟通力
- 执行力
- 推广力

当然，这并非意味着所有的产品经理都需要在上述5项能力上都达到满分的程度，甚至说某些人即使具备了上述5项能力当中的全部，也未必能够将产品做好。在采访著名的软件大师Martin Fowler时，当他谈及缘何能够成为软件大师，并成功推广“敏捷”这一产品的时候，他的回答是：“Luck, luck and luck!”由此可见，运气在打造一个成功产品的要素中扮演了很重要的角色。我们能做的，是在好运到来时能够把握住机会。

下面，让我们分别看看这五项基本能力所覆盖的范围以及它们在做产品当中的价值。

规划力

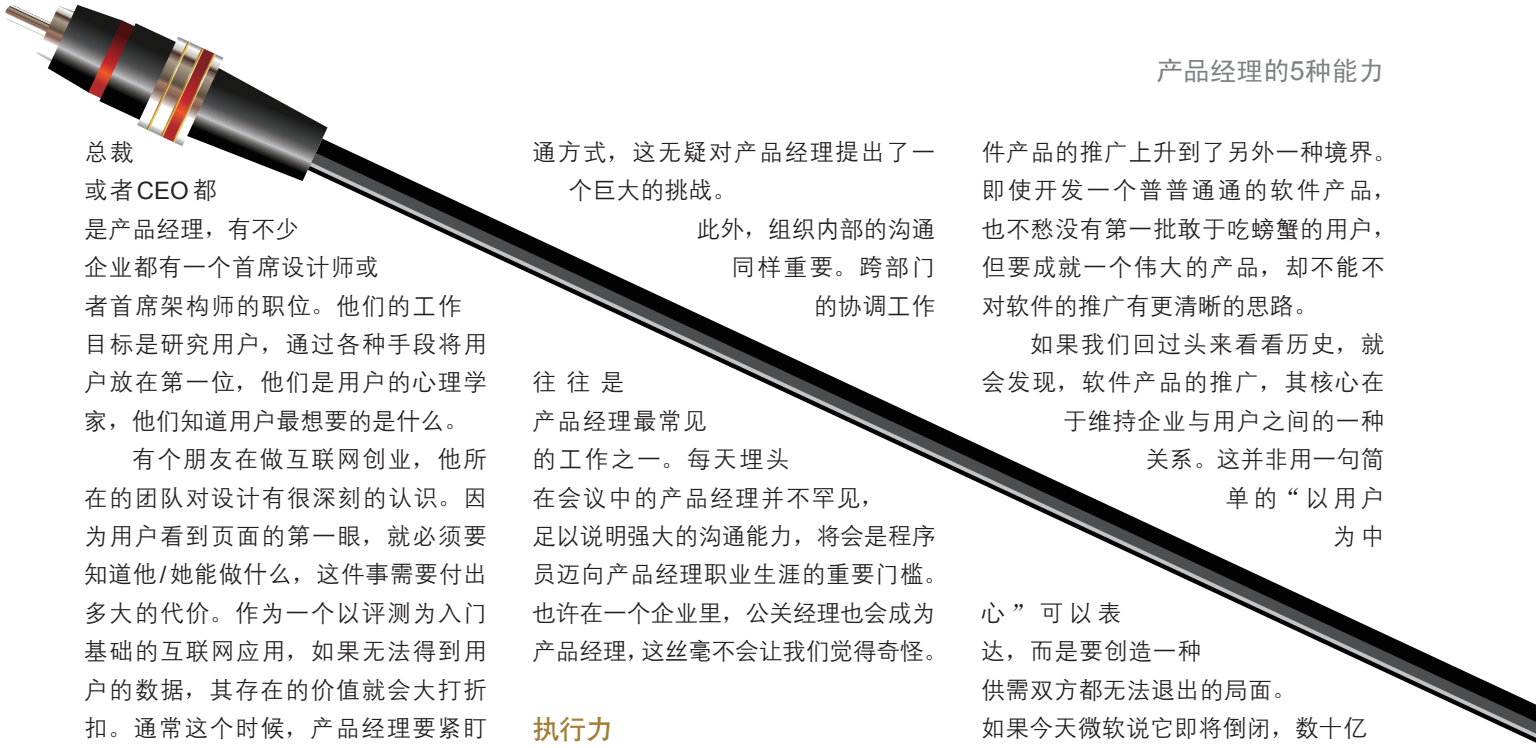
这是一个产品经理较高层面上的能力，而具备这种能力的产品经理，通常会具备很多思想家的特质。从产品外部体系上，规划力代表了产品经理对整个价值市场的认同，对企业产品线的布局，对自身产品的定位以及对每一款产品的发展思路。而在企业内部看来，它所代表的含义还不止这些，它包括对产品研发体系的设计，对资源协调制度的制定，对各种突发事件的把控等。在规划力上具备很强意识的产品经理，通常是产品的灵魂，也是技术团队崇拜的偶像，在一个软件企业或组织中，总裁、CEO等职位最容易成为这样的产品经理。

如果我们看一看今天软件市场上的很多产品就会发现，事实上，很多企业最大的产品经理是这个公司的实际决策人。很难想象一个产品在企业中得不到资源，将会如何发展下去，而事实上也只有整个公司的决策者们了解哪些产品对于他们的公司来说是最重要的。如果一个企业的生存状况，是通过其核心产品的市场状况来反映的话，那么这个产品一定要有比企业内部其他产品更具优势的资源。这一点对大公司来说尤其是这样，数量众多的部门，如何去做有效的权衡与取舍，是大产品经理的必修课。

今天，我们看到众多软件公司高喊的各种口号，其实就是产品经理们为争取更多市场资源的有效印证。SOA、SaaS、S+S、Web 2.0这些概念，无一不是为产品服务的。而为了达到这一目标，还要求企业有节奏、有步骤的一步一步迈向这些概念所提倡的目标。更有不少产品经理，为了达成既定目标，制定出种种制度，比如微软的里程碑制度，就是在这种强大规划力下的产物，从而进一步确保产品的研发和市场投放有序进行。

设计力

事实上，并不是每个软件企业的



总裁或者CEO都是产品经理，有不少企业都有一个首席设计师或者首席架构师的职位。他们的工作目标是研究用户，通过各种手段将用户放在第一位，他们是用户的心理学家，他们知道用户最想要的是什么。

有个朋友在做互联网创业，他所在的团队对设计有很深刻的认识。因为用户看到页面的第一眼，就必须要知道他/她能做什么，这件事需要付出多大的代价。作为一个以评测为入门基础的互联网应用，如果无法得到用户的数据，其存在的价值就会大打折扣。通常这个时候，产品经理要紧盯用户的每一个行为，确保他们有耐心完成我们所期待的每一个动作。

耐心和细致在这项能力当中占据了极为重要的地位。今天我们看到很多软件产品界面上的元素，甚至经历过数百次不断的调整和修改，如果没有这些耐心细致的反复打磨，也许今天我们还停留在命令行软件的时代。无疑，软件世界中美好的一切，都不能不归功于产品经理。

沟通力

在不少企业里，新产品替代旧产品，逐渐成为企业核心竞争力的案例也常常见到。最初，这些产品经理没有多少骄人的权力，但他们却需要承担产品成败的责任，这种产品经理有些像外交家，他们能够通过各种方式有效地取得产品发展所需要的各种资源。

无论是组织内部还是组织外部，沟通都是很多技术人员并不擅长的一件事。这一点在需求工程的过程中显得尤其明显。例如与客户的沟通，软件工程的专家们发明了用例、用户故事等方法，来试图强化产品开发与用户之间的关系。这些方法和工具，今天已经成为大多数产品经理用来沟通的工具了。当然，没有什么比产品经理本身就是业务专家来得更直接的沟通方式，这无疑对产品经理提出了一个巨大的挑战。

此外，组织内部的沟通同样重要。跨部门的协调工作往往是产品经理最常见的工作之一。每天埋头在会议中的产品经理并不罕见，足以说明强大的沟通能力，将会是程序员迈向产品经理职业生涯的重要门槛。也许在一个企业里，公关经理也会成为产品经理，这丝毫不会让我们觉得奇怪。

执行力

建立敏捷企业今天似乎已经成为很多经理人的口号。当然，之所以称之为口号，是因为人们并没有完全实现它。真正敏捷的企业，像训练有素的军队那样行动，臂如指使。这必须建立在有效的执行以及快速的行动上。

市场的博弈，从某种程度上说对企业的挑战就是这一点。在面对强手如林的竞争环境下，很多企业甚至将执行力列在核心竞争力上。很多人批评今天的企业家用近乎军事化的管理来管理企业，然而很多成功的企业确实在用这样的体制开发产品。对应在软件企业当中，尤其以开发部门为要。产品能否按时交付，很大程度上反映了产品开发团队的战斗力，也很大程度上反映了产品经理的执行力与决断力。

正因为有太多臃肿缓慢的团队，限制了软件产品的发展，才导致很多天才般的技术最终被打败。因此，不少企业的产品经理，直接就是开发项目的项目经理，尤其是那些本身就充满了创意的产品，以最快的速度交付它，是实现竞争力的关键。

推广力

互联网的出现，彻底改变了软件的传播方式。当零售商店与盗版小贩不再成为软件传播的唯一途径时，软件产品的推广上升到了另外一种境界。即使开发一个普普通通的软件产品，也不愁没有第一批敢于吃螃蟹的用户，但要成就一个伟大的产品，却不能不对软件的推广有更清晰的思路。

如果我们回过头来看看历史，就会发现，软件产品的推广，其核心在于维持企业与用户之间的一种关系。这并非用一句简单的“以用户为中心”可以表达，而是要创造一种供需双方都无法退出的局面。如果今天微软说它即将倒闭，数十亿的用户都不会同意，因为架构在微软技术和平台上的产品将失去最后一层的保护，不会再有任何人为它们负责；同样，如果IBM今天宣布倒闭，全世界的银行和制造业等企业都会伸出援手，因为他们深刻地明白，双方都无法从这种局面里退出了。

在互联网时代，你会如何推广你的产品？互联网也好，销售渠道也罢，这些都是那些产品市场经理运作的手段，真正要将产品推向更广阔的市场，不仅要具备对市场的敏感、独到的眼光，更需要有产品的大局观。

写在最后

其实，一个产品的生命周期是可以预期的。它基本上被分成三个部分，即策划、实施和营销。如何有效把握这三个阶段当中的每一个部分，会根据不同的产品策略以及不同的企业行为来决定。但通过访谈，我们却了解到了另外一个被普遍认同的共识：心态问题。

产品经理应该具备什么心态？如果说积极向上是一种心态，那么未免显得有点敷衍。然而，这并非本文的讨论范围，希望那些有志于成为产品经理的技术人员在掌握了强大的技能之后，能够更好的把握心态，做出震惊世界的产品。■

XOOPS 发布有期：谁说开源不计划？

■ 文 / 姜太文 高瑾

1. 写在前面

近年来，开源软件在中文用户中日渐流行。尤其是今年，随着微软黑屏时代的降临，很多用户终于抛弃坚守多年的盗版方针，开始发现和探索开源软件。虽然开源进入中国已经相当长时间，可是在普通用户甚至IT业界人士中，仍然存在很多认知误区。这一点是所有参加开源项目的人们应该注意并努力改变的。

这些误解中，关键的错误是对开源项目管理的理解。比如，很多人以为，一个开源项目就像从长江源头开始漂流的一叶轻舟，兴之所至，漂到哪里算哪里，停留和出发都仅仅是偶然的。这种误解造成的严重后果就是对开源软件某种程度的怀疑。确实，一群人毫无计划地聚在一起、纵横捭阖地做出来的东西，听起来虽然潇洒，可是恐怕不够严谨细致。

其实，计划对于开源项目来说，至关重要。可以说，开源项目如果没有计划，必定失败。

2. 为什么认为计划是开源项目最重要的？

开源项目与公司体制下的商业项目相比，在开发管理模式上缺乏强制性的组织管理机制，从而决定了开源项目计划的重要性。开源项目一般都是由来自不同地方的参与者在志愿方式的基础上进行远程管理协调和开发测试，在开发过程中缺乏面对面及时有效的交流沟通。开源项目参与者的工作需要依靠统一的计划保证进度的统一性。如果没有灵活完善的项目计划，基于松散组织的开源项目的开发必将会陷入各自为政的混乱状态，无法保障项目的健康发展。因此，如何设计并制定灵活有效的计划对于开源项目特别是有多人异地参与的较大规模开源项目来说是至关重要的，也是随着开源项目的发展壮大而不断更新的研究课题。

3. 既然计划是开源项目最重要的方面，那么开源产品经理与其他产品经理的不同点在哪？

在商业公司或团队中，产品开发计划由产品经理负责制定并监督执行。在开源项目中，不存在“产品经理”这个职位，其功能职责一般由相关团队负责人共同完成。

商业团队的产品经理根据公司一定时期的发展策略和相关规划设计产品功能并制定开发计划，并依此配备相关人力资源，保证产品开发的进度和质量。而开源项目的项目开发计划则主要根据主开发和架构师的前瞻规划并参考来自用户社区的需求制定一个时间段的项目设计和开发计划。同时考虑到开源项目参与者组织模式和开发模式的特殊性，开源项目的计划制定更侧重于功能计划，难以对时间进度作强制规定。同时无论功能计划还是时间进度规划都要根据可用开发资源即开发者的时间可用性和不断变更的社区功能需求作及时更新调整。

4. 具体来说，开源项目的计划一般应该怎么做？

一个成功的开源项目是一般由两部分组成，开源软件本身和由开源软件开发者和用户组成的社区。

具体来说，主开发员和架构师制定项目的整体开发路线图，并制定任务分解方案；核心程序员按开发路线图制定并分解核心功能开发计划；普通参与者可根据细分的计划方案制定个人开发计划，与系统开发步骤保持一致。社区管理团队制定社区管理方案和年度活动组织计划，每个功能团

队根据社区年度计划制定自己团队的工作计划，与相关团队协同开展工作。

5. 作为Sourceforge十佳项目的开源项目，门户管理系统XOOPS是怎么做的？

XOOPS是世界上流行的一个开源门户系统，产生于2000年底，至今已有八年的开发历史。XOOPS系统分核心层和模块两部分，核心团队负责核心层功能的设计和开发，功能模块由第三方开发者或公司提供。

核心功能如何定义？XOOPS第一个版本的故事

与其他大部分开源项目一样，XOOPS是由一群有共同开发兴趣的开源开发者和爱好者共同发起的，在phpnuke基础上对原系统做了重写，经过大约一年的设计开发在2001年底发布XOOPS第一个公开版本。

XOOPS社区建立的TODO list

XOOPS是一个纯社区支持的开源项目，无论是开发者还是社区管理者都是由开源参与者自愿组织开展工作。自项目成立伊始，就依托于世界上最大的开源项目管理平台Sourceforge，采用Sourceforge提供的项目管理工具和自己的社区网站对项目进行规划管理。普通用户在论坛或文章评论里对XOOPS核心或模块提出自己的改进建议和功能需求，由社区志愿者协助整理到Sourceforge的功能需求列表。考虑到普通用户的使用习惯，在社区网站的Wiki里也建立相应的功能列表，便于用户查阅修改。在XOOPS各地分语言支持站也建立相应机制，由各支持站大使负责向总站汇总功能需求。核心开发团队则根据XOOPS核心开发计划和社区功能需求定期制定发布XOOPS开发路线图和详细的TODO list，并定期报告TODO list完成状况。

如何吸纳有经验的代码贡献者以及他们在项目中的角色

XOOPS的代码管理早期采用Sourceforge提供的CVS，后来改用SVN系统。XOOPS的核心代码由核心开发团队维护。主开发负责整体架构设计和任务划分，每个核心开发员根据自己的特长和时间负责相应部分的代码开发和接受并验证其他开发者提供的代码。对于长期提供核心代码的开发者，经过一定时间的培训考核后赋予相应的代码提交和维护权限，逐步成为核心开发员，并帮助其他开发员参与核心代码的开发。同其他开源项目一样，XOOPS在某些特定时间段可能会存在两个甚至更多开发版本同时存在。这种情况下，主开发员会指定某个核心开发员总体负责相应版本的开发和版本发布。

在这种自愿参与的机制下，循环往复，维持了XOOPS核心团队的稳定和不断成长。

版本升级的功能定义及裁剪

XOOPS的发展轨迹是由大的版本和不同水平细分的小版本发布构成的。到目前为止，最新公开稳定版是2.3，内部开发版是3.0。

大的开发版本主要涉及到架构的演进和模块开发重要接口模式的调整。这类新版本的设计规划一般由主开发员支持，由核心团队成员共同讨论制定，并结合社区反馈意见作相应开发进度调整。该类版本的功能和特征主要由主开发员和核心开发团队制定并发布给社区。一个大版本的发布一般要分别经历多个Alpha、Beta和RC发布，最后发布最终版，开发周期一般在一年甚至更长。

中间版本主要在维持当前架构和模块开发接口的前提下，对功能和性能进行改进，并根据情况增加调整新功能。这类功能调整和增减大部分来自社区的需求和反馈。发布一般要经过一个或多个Alpha/Beta版，然后进入RC版。在RC版和最终版之间维持至少两周的测试反馈期，以保证社区测试的充分性。每个版本开发周期一般是三个月或半年。

小版本主要是bug修正，一般不涉及功能的调整。每个完整发布一般由一个或多个RC版和最终版构成。RC和最终版间隔遵从两周的时间规则。但是在出现安全问题的情况下，核心开发团队会尽快发布补丁，不再遵从时间间隔规则。

计划管理bug和日常工作

与需求管理类似，XOOPS团队采用Sourceforge的Bug Tracker和社区网站的论坛对管理bug报告和修复。用户在论坛里报告XOOPS的问题，由社区志愿者协助整理到Sourceforge的Bug列表。核心开发人员会及时分析Bug List，在SVN提交修正并在Bug Tracker里更新bug状态。在下个版本发布时对相关bug做必要的说明解释。

XOOPS开发团队还提供了安全问题报告机制，主开发员负责安全问题的分析，并及时作出修正，必要时会采用临时应急处理措施。

社区维护以及反馈的规划怎么做？

XOOPS组织由项目负责人即主开发员和社区负责人领导下的数个功能团队构成，分别负责项目开发和社区组织管理的相关工作。团队成员根据用户意愿和精力不定期调整。

社区维护团队的工作主要以论坛维护的方式体现，志愿团队在社区帮助新用户了解学习使用XOOPS系统，并整理用户报告的问题和提出的功能需求，协助开发团队做好开发及测试规划。

6. 无冕之王的开源产品经理小结

开源项目“产品经理”对开源项目的成败是至关重要的，是一个开源项目从项目策划到开发及应用顺利进行的保障。■

作者简介

姜太文，XOOPS主开发员，项目负责人，XOOPS中文社区管理团队成员。

高瑾，人文与社会网站（wen.org.cn）主持，耶鲁大学比较文学博士生，XOOPS中文社区团队成员。

TechExcel 的“灵魂”缔造者

■记者 / 付江

对于软件类公司，如果没有开发团队，必然做不出产品，但倘若缺少了设计团队，产品也会失去“灵魂”，即使最后勉强将产品做出来，也很难获得用户的最终满意。采访一开始，TechExcel CEO兼首席软件架构师周铁人给记者这样描述了他对产品经理的理解。

产品经理决定产品“灵魂”

周铁人回忆，在TechExcel刚刚成立时，只有5名员工，参加了开发第一款产品DevTrack的全过程。他身兼数职，既是CEO又是产品经理，不但参与了前期的代码开发、市场调研、客户需求搜集和整理，更是设计了DevTrack的全部功能，甚至是产品界面，而其他的工程师则分别负责数据库、源代码模型的工作。三个月后，周铁人便将主要精力放到产品设计上来。

周铁人始终坚信，“设计力”是一个合格产品经理需要具有的众多素质中最重要的一项。对于普通用户，他们并不在乎一款产品是如何做出来的，而只关注产品能否解决他们的实际问题，产品是否及时增加了他们需要的新功能，过去不实用的功能在新的版本中是否已经取消掉了，产品界面是否友好，是否便于操作使用等。而在TechExcel也一直有一个非常好的传统——“设计先行”，就是说任何一项新的功能在通过产品经理的讨论之前绝不会递交到开发团队手中。在TechExcel内部，产品经理和开发经理的任务有明确的分工，产品经理的主要职责是保证产品拥有正确、恰当的功能，而开发经理的职责在于新功能的开发和功能的改进。

软件领域能够在北美创业成功的华人，并不是都像周铁人这么幸运

TechExcel产品经理工作模式

经过多年发展，到目前TechExcel已经形成了3条产品线，分别为ALM、CRM和ITSM。每一个产品线都有多款产品，现在一共有10位全职的产品经理，以及周铁人自己。作为公司的CEO和首席软件架构师，在公司发展到一定规模后，周铁人不可能再将过多精力关注在各款产品的代码及一些具体功能上，他更多的是从公司产品的整体性和战略性来思考问题，将产品经理归纳提炼出的用户需求和功能设计做进一步的提升，从众产品经理贡献的思想中找到既能满足客户整体需求，又能促进公司产品各方面提升的切合点。“提供一个广阔的平台让每位产品经理充分发挥他们的才能是我的工作之一。”周铁人说道。

10位全职经理也会有具体的分工，每一个产品经理会全力负责1~2款比较相似的产品。他们的学习和工作背景可以分为两个大类。其中有5位都是非计算机科班出身，有一些计算机知识理论，但都不曾有过正规的开发经验。这5位过去主要都是从事市场开拓、客户服务，甚至是销售，也就是说他们积累更多的是在一线和客户直接打交道的经验。非科班出身的好处是在整理客户需求的时候可以不受编程思维的限制，更好的站在普通用户的角度与客户交流，在提炼需求的时候脑海中首先出现到的不会是开发工具或是功能实现模式，注重客户对产品的反馈意见，因此他们可以更好的配合销售人员与客户进行沟通，将客户的需求归纳提升后，将需求进行量化管理，并把需求与设计和文档相关联。

而另5位产品经理都是在国内最

好的大学毕业后又到国外读了硕士或博士的计算机科班出身，他们来到TechExcel在开发团队一线做了近两年的开发，拥有丰富的开发和架构设计经验，对相关的开发工具和应用模型非常熟悉，因此他们能更好地和开发团队合作。在产品设计上，他们更侧重将设计具体化，将需求量化并形成比较完整的概念产品。协助开发经理将需求呈现给开发团队，并周期性的协助开发经理对产品开发进度进行评估。这5位产品经理有时候也参与数据库模型和产品架构设计，但这并不是他们考核的指标，会有专门的开发团队和架构师团队来做这些工作。在TechExcel，设计和开发的过程遵循SpecDD(Specification Driven Development，规范点驱动开发)理论。作为周铁人自己创立的一种敏捷的开发方法，它对敏捷方法中的SCRUM和FDD(Feature Driven Development，功能驱动开发)进行了融合和改进，在开发的过程、文档、团队和需求管理多种因素中寻求一种平衡。这也是TechExcel一直坚持使用的开发方法，其中在需求部分，都要进行量化和系统的管理。

在谈到国内外产品经理的异同时，周铁人认为两者在思想上有很多共通之处，例如都重视客户需求、推广等，但对产品目的的理解上还是存在区别。在TechExcel，并不力求原汁原味的满足客户提出的各方面要求，这实际上也是很难完全实现的，甚至有时会忽略公司产品的完整性和客户需要解决的实际问题是什么。相比较而言，国外的ALM厂商、产品经理包括用户相对来说更加成熟，更注重产品的完整性和长远性。目前几乎全球最大的游戏公司都是TechExcel的客户，但即使是这样的客户提出的需求，也很难保证可以百分之百的满足客户需求，但通过双方的充分沟通总能找到一种最高效的解决方案。现在，TechExcel已经和全美10大游戏公司的7家建立合作。在中国，盛大公司和巨人网络这2家顶级的游戏开发商，也相继成为TechExcel客户。

产品定位决定推广策略

产品经理在考虑和设计产品时，不要只看到一个介质，一个最终成型的实物，一个将功能组合在一起的集合体，而是要赋予这些介质能够传达下去的灵魂，需要对自己的产品有准确的市场定位和将来可能采取的推广手段。在考虑产品定位的时候，产品经理应多问自己："我这个产品是做给谁的？"，"产品的目标用户在什么地方？"，"目标用户都有什么特征？"，"我是否有竞争对手？"，"竞争对手的产品有什么特点？"，"我应该如何扬长避短？"等等。结合到TechExcel，虽然当时在ALM领域，Rational和Borland已经非常强大，但TechExcel还是在以下几点发挥了自己的特长：采用了更加友好的产品界面，提供更强大的工作流引擎，可以巧妙的处理复杂的团队协作和生命周期管理；产品具有高可扩展性，适应于不同规模的团队应用。拥有了明确的市场定位之后，在做推广的时候还要多考虑这个群体的心里感受是什么，一定要考虑的全面一些，尽量把推广策略的细节想充分。

TechExcel在公司发展的不同阶段，采取的产品市场拓展手段也各不相同。公司产品刚上市的时候，由于缺乏市场经验，也缺乏市场经费，TechExcel没有做太多的市场工作。客户很多都是由一些老客户推荐过来的，他们到网站下载评估了产品，经过试用最终都非常认可TechExcel的DevTrack，并最终购买了产品。

周铁人认为，厂商们总是想在自己的产品上增加更多的功能，产品固然重要，但其实更紧迫的是去了解一线客户们更关注的是什么。他自己有一段非常有趣的经历，2005年的时候，自己曾到TechExcel的一家老客户First American公司参观。公司老总和CTO接待了他，带他参观了公司工作区。这家公司五万多人，做IT技术支持的有一千多人都在用TechExcel产品，墙面上贴满了用TechExcel完成的报表。在这之前周铁人从未想过会有公司将用TechExcel产品完成的报表贴满在墙上，这让他非常感动。在与员工交谈中，周铁人了解到，他们最喜欢TechExcel产品的地方在于：易用性和处理速度。他们过去使用的是一家当时市场份额最大的公司提供的报表产品，生成一张报表需要花几分钟的时间，这时员工往往可以去泡一杯咖啡，回来报表却还没生成。但使用TechExcel的产品后，他们泡咖啡的时间没了。

回到公司后，周铁人将TechExcel产品的处理速度和易用性提到了更为重要的地位。并且，在那段时间，TechExcel在自己的广告和市场活动中纷纷加入了这句广告语"It's not coffee time！"也让更多的用户了解到了TechExcel产品的性能，公司产品销量也节节高升。

展望将来，周铁人坦言自己常常在思考TechExcel公司未来五年的发展规划，他认为客户关注的还将是软件开发的产量和质量，速度快，产量大质量高的产品仍将受到欢迎。在业务流程自动化和过程优化这个领域，对软件工具会有很大的需求。一款优秀的ALM工具，不仅可以适用于敏捷开发、迭代开发、CMMI、瀑布开发等多种软件开发模型，还应该以客户价值为中心，不断完善产品本身的性能，使其功能更加强大、可扩展、易于维护，否则就无法获得持续的市场成功。

"而Web 2.0技术和应用也会向前迈进一大步，并且大面积加入到企业软件应用中，渗透到企业业务流程管理中。将Enterprise 2.0运用到ALM领域，能帮助越来越多的软件开发团队采用标准的开发管理工具，进行规范化的软件过程管理，在这个过程中，TechExcel愿意做一个先驱者。"周铁人兴奋地说道。■

LiveBOS 的深度需求工程

■ 记者 / 周至

一般地来讲，一个平台级别的产品，开发团队动辄几百人，而且耗资巨大。但是顶点软件CEO严孟宇所率领的LiveBOS开发团队，通过多年的摸索和实践，只用了较少的人力和成本，却高效率、高质量的完成了LiveBOS这款对象型业务架构中间件及其集成开发工具。这款工具不仅实现了以业务模型建立为中心，直接完成软件开发的创新软件开发模式，而且适用于基于WEB的专业应用软件与行业大型应用的开发。

那么，究竟是什么样的原因，让这个原本基于C，C++的GUI开发的团队成功转型为基于JavaEE的WEB开发，并高效率、低成本地开发出如此大规模的软件产品呢？记者此次特别对顶点软件CEO严孟宇先生进行了一次专访，请他从开发LiveBOS这款产品的切身经验谈起，对他一直以来所做的工作——产品经理这一职位作以诠释。

顶点软件的严孟宇承担了平台产品经理的角色

深度发掘需求，做好企业产品

严孟宇在LiveBOS这款产品的开发过程中，基本上负责的就是一名产品经理的工作。具体的来讲，主要包括三个方面：为产品定位的工作，即对将LiveBOS做成一个什么样的产品作以决策；对产品的具体特性做出一定的要求；对技术经理做出来的成果进行评价，并针对评价提出相应的改进意见。具体谈到产品需求分析方面所做的工作，他谈到了以下几点：通过与客户的沟通，定义产品的规格；定义产品的主要界面风格；定义产品的使用环境；定义产品的关键特性；与技术经理共同定义技术的基本架构等。

而这做几方面工作的过程中，严孟宇一直秉持着同一个理念，那就是一切从需求出发，只有满足了客户需求的产品，才是一个好的产品。

例如，在为LiveBOS这款产品进行定位的时候，他就是完全基于客户的需求，以及他对市场和用户的理解和认知。

由于他所定位的市场和用户主要是行业运用开发商与SI，所以他深入地分析了这些人的需求所在。首先，行业运用开发商与SI主要从事的是一个基于WEB的开发过程，而这个开发过程的特点就是开发周期长，过程相对繁琐。而恰恰由于这一点，决定了这两种用户的核心竞争力并不是是否拥有一批掌握了某种开发能力的技术团队，而是一种行业专家的概念，是对行业的一个深刻的理解。所以，这类公司比较容易接受的产品是一个行业应用的解决方案，而不是一个单纯的开发工具。

于是，他提出了业务架构中间件的概念。他并没有将这款产品定位为一个简单的辅助开发软件，而是一个业务的中间件。即通过一个Xml文件的模型就把设计首先变成了一个应用，而不单单是生成简单的代码。

产品经理：优秀的需求分析师

谈到一个产品经理所必须具备的基本素质，他认为主要应该有以下

几点：

1、优秀的需求分析能力。

2、总结与分析的能力。

3、对技术的理解，对技术发展趋势的一个理解。因为这个产品本身就是面向开发人员的，所以对开发技术的发展方向上的理解与判断能力非常重要，即对技术发展的趋势的判断能力非常重要。

4、学会借鉴。即对国内、国际上技术的理解和分析能力。在LiveBOS这个产品几年来的开发过程中，他研究了国内外大量的同类型产品，对他们一些特性的发展趋势也都做了深入的分析。他认为，只有这样才能够把握产品开发的正确发展方向。所以，产品经理一定要时刻注意到产品发展方向的变化，从而进行正确的决策。

此外，作为产品经理，也应该参与一些技术选型工作。产品到底选用哪一种技术架构，实际上也是决策的一个过程，因为使用不同的技术架构，会在很大程度上影响到客户群。

而在这些基本的素质之中，他指出最重要的就是需求分析能力，也就是说，产品经理应该同时是一名优秀的需求分析师。

产品经理的需求分析工作应该在整个开发流程中一直存在。他认为，产品经理应该在实施过程中分步骤地去指导项目组成员，定期的去研究、分析一下他们的开发成果，检查是否跟预先对于产品的定义相一致，也就是是否与用户最初的需求相一致。当出现不一致的情况时，就需要采取重新再讨论的措施，或者经过分析之后，判断是需求需要调整还是运营的产品需要改进。

此外，他提到了在开发LiveBOS这款产品的过程中，需求的定义是一个不断滚动的变化过程，而不是那种事先完成非常规范和完整的设计之后才进行开发的。所以他们的团队也融入了一些敏捷开发的概念，在不断的实践中确定真正的需求所在。

好的需求源于沟通

严孟宇认为产品经理在完成需求工作的过程中，沟通能力是最重要的。而且，这里的沟通不仅仅是团队内部的沟通，更多的是与用户的沟通。

他举了一些例子来阐述这一点：比如，LiveBOS开发团队在产品开发的过程中，就会更多地采用结对开发，即让同一组的开发人员之间加强沟通，让他们对于需求的理解尽量做到一致。

另外，在产品做到一个“里程碑”的时候，他们还会请他们的用户与开发组的成员进行面对面的一个沟通，开一个产品介绍会，让这个产品在一个小的最终用户群里做内部推广，让用户进行试用。最后，他们可以通过这样的活动及时地得到一个反馈的需求信息。而产品经理就会根据这个信息，做一些需求上的变动从而改变相应的开发计划。

规划产品周期，稳步适应用户需求

严孟宇认为，合理的规划好产品的生命周期，也是产品经理所必须做到的。这一点的实现是一个技巧，一个合理的产品周期，可以稳步的满足用户的最终需求所在。

一般来讲，在开发LiveBOS这款产品时，他会在三到六个月内根据用户的需求，为产品定义一个重要的特性，并对产品进行小的版本更新。而通过这样一个小的周期，既可以快速的实现这个新的特性，又可以分析这个特性的出现是不是满足了用户的真正需求。而大的版本设计往往是针对用户已经适应的特性，在这些不同的特性之间进行重新编码、整合的过程。简单地说，大的版本更新就是把已有的稳定的特性保存下来，对零乱或逻辑性不够强的新特性进行整合的过程，夯实基础，满足用户的需求。

而且他强调，在一个新的特性研发出来之后，他会特意的去维持一段稳定的完善期。即使这时有了一个新的特性需求，也会控制住，不会去做这个新的尝试。也就是要控制版本更新的一个节奏。与此同时，完善内部的相关文档，进行一些细节的调整。

此外，正如前文所提到的，在每一个小版本更新发布的时候，LiveBOS开发团队都会开一个特性的介绍会，或者新版本推广会，时时倾听用户的心声。

一名产品经理对未来的展望

在产品推广LiveBOS这款产品的过程中，严孟宇也曾经遇到了一些困难，因为一些企业已经有了自己固有的开发模式，让他们一下子接受这个产品的确有一定的难度。但让他欣慰的是，已经有一些企业正在慢慢地接受他们的产品。而且，很多最终端客户在看到了这个产品之后，觉得LiveBOS开发团队虽然只拥有很少的技术人员，但是也能开发出大型的高效软件来，那么这个产品一定能够给他们带来很大的帮助，并且能够提升他们产品的运营水平。

谈到对未来的展望，严孟宇对于未来三年内的计划是这样的，在产品功能方面，重点让产品形成从测试到流程的控制，包括软件开发过程的控制这样全周期的支持，使整个特性得到完善，当然，这一切也都基于用户对于产品全周期支持的需求。另外，业务方面的目标就是在国内能够得到一个大规模的推广、使用。

至于进行这些推广的资源，他准备从两方面进行获取，即人和资金。人的方面，他会从应用软件的开发商团队中抽调一些在应用软件开发过程中拥有优秀的技术，而且表现出比较好业务敏感性的技术人员来补充LiveBOS开发团队的力量，还有就是通过社会招聘；资金方面，顶点软件本身每年都有比较好的利润和现金流，同时也在申请一些政府的创新资金，在未来的几年里会考虑引进一些外面的资金去做更大发展，比如风险投资等等。■

创意搜狗输入法设计

■ 文 / 马占凯

随着市场份额的增长，搜狗输入法已经获得了用户数量上的成功——现在每两个中文网民中，就有一个人在使用搜狗输入法打字。可以说，它是一个从创意灵感到产品诞生到应用的很好的案例。作为搜索引擎输入法创意的提出者与推动者，我经历了这个故事的全过程，从灵感的诞生并将创意推销给副总裁，一直到做搜狗输入法的产品设计的全部经历，也成为我宝贵的人生阅历。应《程序员》邀请，以搜狗输入法为例，在这篇文章中与大家谈一谈什么样的技术能转化为成功的产品。

从触动到行动

也许很多人会关心搜狗输入法的原始灵感来源。那还是我毕业后的第一年，正从事机械行业，是一个典型的对互联网感兴趣的用户。每天下班后就是玩电脑、看IT新闻、上网。当时我使用的主要两个产品是紫光拼音输入法与百度搜索。

日积月累的网上冲浪，让我发现紫光拼音当时的一个严重问题：很多常用词没有在输入法中建立词库。比较典型的几个词比如诺基亚、泰坦尼克号、张含韵等，几乎都要一个一个从词条中选择。而另外一个在搜索引擎的发现却让我觉得非常受用，那是百度网页搜索中一个特别好用的功能：拼音识别。例如在搜索文本框中输入“taitannikehao”，百度会给出“泰坦尼克号”的提示。

经过实验，我发现这个功能无比强大，几乎任何拼音都可以识别为中文。当我无法接受这样不顺利的输出文字的时候，便很自然的想到为什么不能把百度搜索拼音识别功能用在输入法上呢？这样几乎任何词都可以打出来。

之后，我带着这个创意，经历了被百度拒绝，到后来说服搜狐做输入法再到现在的广泛应用，但是这个最初灵感诞生的过程是最重要的吗？我相信每个人都经历过类似的事情，遇见很多的困难，但是真正想办法去解决困难的人却很少。可往往成功的产品都是在这样的小困难中诞生。这给我的启示是：当你在用电脑或者互联网时，遇到了什么问题？是很多人出现的问题吗？是否可以解决？怎样解决？解决后的效果会怎么样？是否会让更多的用户得到满意？经过详细拷问自己，深入思考，你就会发现产品需求在哪里。

大众应用类产品的一个最重要需求点就是实用。如何把实用发展到长期应用，这是问题的关键！也就是我们所谓的用户粘性度。我把产品粘性定义为：这一次用户用了这个产品之后，下一次用户还要用，就证明这次使用产生了粘性。你的创意是否让人们得到了实际收益？如果设计的产品仅仅是让人们觉得挺炫、挺好玩的，不考虑用户的实际需要，那这样的产品是不会被人们所使用，也不会获得成功，只有把握了用户的需求和超过用户需求的体验，产品才是成功的产品。

有准备的成功

任何产品在没有诞生的时候，都会遇见不同的意见，坏的大于好的；任何产品在没有得到个别用户认同的情况下，你要想尽办法丰富完善自己的创意并通过各种途径去接触更多有价值的人。我当然也不例外。

我带着我的创意点来北京后，先给百度写了4封信，但没有得到肯定的答复。在被拒绝后，我凭借给搜狗提的100个小建议的30页文稿而有幸进入搜狐。我才有机会面对面与当时负责搜狗的搜狐VP老王（王建军）演示我做的第1个关于搜狗输入法创意的ppt。我还清晰记得第1个ppt的内容就是阐述目前的现状，企业都在抢工具栏、地址栏（当时很多企业都在拼命推广工具条类的插件），与其这样，不如我们去做一个带工具栏的“搜索”输入法，一方面我们占领了工具栏，另一方面用带搜索的输入法抢占这个市场。奇异的想法很快就打动了老王。当时大家都希望立即收购紫光输入法，以最快的方式占领这个市场。但是过了段时间之后，紫光拒绝被收购，输入法的事情又被搁置起来。此刻的我重燃激情，继续游说，我不相信这个创意就此沉没。

尝试了失败之后，我发觉说服别人要有更深层次的理论，不能简单的靠嘴，要用数据去说服领导。此后，我住在城中村的平房中收集各大软件下载站的输入法下载量。统计完毕之后，我被统计数字所震撼，三大软件站所有输入法的下载量是1亿次。由此推断，整个互联网的输入法下载总次数大约在2亿-3亿次。这个数字太恐怖，除了QQ没有任何其他软件能够达到如此大的下载数量。输入法的需求非常大，任何一个上网的国人打开电脑都会用到汉字。那么需求已经有了。我又统计了一下03、04、05年市面上现有输入法的更新版本，答案是几乎没有任何更新。这两份数据证明了我对搜狗输入法的决断是正确的，更加坚定了我做下去的决心。

我把新的想法做了第2个PPT，又给老王演示，时间一秒秒溜走，就在这无形的时间中我知道我已经打动了老王要去做这个事情的决心。我终于把输入法的创意推销出去了。那一刻无法用“喜悦”、“高兴”来形容。

解析产品经理

如果你想做一款成功的大众应用类产品，应该选择什么样的产品开发呢？我想从需求层面来说一下，主要从3个方面来考虑：需求人数、需求次数、需求的重要性级别。需求人数决定产品能有多少潜在用户。例如在五笔之后几乎所有的笔形输入法注定不会成功，因为能够学会用笔形打字的用户数量非常少。如果你的产品非常适合广大的傻瓜用户，你的基数就会非常大，即使只有百分之几的份额，也会有千万级别的用户。需求次数与重要性分别决定了产品被使用的频率与强度，这两个因素如果不够的话，就难以成为一个成功的产品。例如互联网上的富有创意的小产品，玩着还行，但是玩了一次之后就不会玩第二次了，这样的产品无法成为有竞争力的产品。再比如你知道的成功产品几乎都是每天都要用很多次并且都必须要用的产品。

需求层面也仅是一个基础，要想获得最终的成功还有很多因素。在《创新的扩散》一书中有这样的描述：“一个创新产品的扩散速度的五个相关因素：相对优势、相容性、复杂性、可实验性、可见性。”搜狗输入法在这几个因素都处于优势，尤其是相对优势。先入为主的思想使人们对第一个真正好的产品印象深刻。搜狗输入法相对于传统的输入法优势非常明显，过去多年打不出来的词语一次全打出来了，不漏一词。同样是第二代的QQ输入法和谷歌输入法即使追到了搜狗同级别的水平也难以被用户接受了。当相对优势很小的时候，产品的竞争力也就很小。在一个产品立项时就应该考虑：你有没有明显的相对优势让用户来使用？

在输入法发布第一个版本之后，我调到输入法组，负责了搜狗输入法两年的产品设计工作。这个期间我们以大概2个月1个版本的节奏发布了十几个版本，设计了大大小小几十个小功能。在设计这些功能与改进时，主要的一个思路就是实用。这些实用的功能得到了高端用户群的认可与欢迎。这些高端用户群又在传播产品的过程中，起到了良好的口碑作用。《引爆流行》对关键传播用户做了很好的描述：“在互联网的产品传播中，这些用户就是一个班级中的电脑高手、公司的网管、部门的电脑专家。他们评判各类的新潮软件，试用各类功能并把自己认为最好的产品推荐给几十个身边的人。”

产品经理在负责一个产品时主要有两个方面的工作：基础指标与产品功能设计。以输入法为例，搜狗输入法在基础指标的词库覆盖度、新词覆盖度、组词准确率等方面，均处在前列。产品功能的选择，同样参考了需求人数、需求次数、需求的重要性级别这三个因素。使用人数很少的英文输入法我们放到很晚才开发。

精通技术不是必须

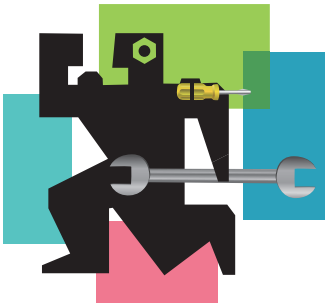
最后，我谈一谈对技术人员能否做产品经理的看法。我认为，精通技术与是否适合做产品经理没有太大关系。也就是说，即使你不会技术，你也未必在做产品经理上有更多劣势或优势。只有在创意、思维、前瞻、策划等方面擅长的人才适合做产品经理，就像很多人成不了优秀的程序员一样。精通技术并不能成为你去做一个优秀的产品经理的指标。是否适合做产品经理只有一个评价标准，能否发现问题（产品需求）、解决问题（产品设计）。每个人都对产品有看法，但感兴趣和在这方面擅长是不一样的。要客观的评价自身，不要浪费自己的真正优势。值得一提的是，产品有很多类别，你如果不适合这一类但是可能在另外一个产品类别上擅长。例如，很多游戏类、社区类、应用类产品是不一样的。但是可以肯定的是做游戏的必定是对游戏痴狂的人才行。

产品经理是一个产品项目组中的专业化细分职位。无论是做技术还是做产品只要做的好，都会有很好的出路，主要看你的优势、兴趣在哪里。一个懂技术的产品经理相对于一个不懂技术的产品经理肯定是有对产品熟悉的优势。如果你平时创意很多，对各家产品的功能了如指掌，思路不偏，看法不固执，自己能够代表典型的傻瓜用户群并且真正的知道怎样把一个产品做好，那么你就适合做一个产品经理。但是从平凡到优秀，从优秀到卓越还需要经过实践检验。■

作者简介

马占凯，原搜狗输入法产品经理。最早提出了搜索引擎做输入法的思路，在进入搜狐公司后建议并推动了搜狗输入法的诞生，随后承担了搜狗输入法的两年多的产品设计工作。由于搜狗输入法的创意来源于马占凯，故网络有传外号“搜狗输入法之父”。

浅谈 51.com 产品设计



■ 文 / 王武佳

在百度百科的“产品经理”词条中，第一句话是这么描述的：“自1927年，美国P & G(宝洁)公司出现第一名产品经理(Product Manager)以来，产品管理(Product Management)制度逐渐在越来越多的行业中得到应用和推广，并且取得了广泛的成功。”

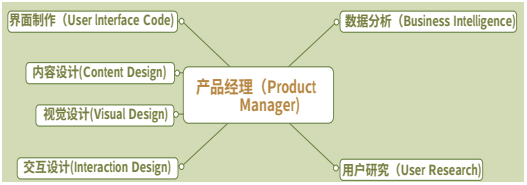
在广义IT业中，产品经理制度虽然已经在软件、通讯设备等行业得到了广泛的应用，却一直没有得到外界太多的关注。而最近“产品经理”一词逐渐热起来，则应归功于它在互联网业界中的流行。

互联网是个年轻的行业。而产品经理则是这个年轻的行业中最年轻的职位之一。在Web 1.0时代，极少有公司设置这个职位。网上的内容，绝大多数是由少数人（网站编辑）来创造、处理和分发。我们大概可以勉强地说，网站编辑就是1.0时代的产品经理。

而随着Web 2.0浪潮的席卷，用户创造内容（UGC）开始慢慢成为互联网的主流。因此，在2.0网站中，设计一个适合用户创造内容的平台尤为重要。而产品经理正是这样一群主导着网站平台设计、建设的人。由于产品经理是产品需求的最初提出者和主要设计人，他们的设计最终决定了网站将以这样的界面、功能、交互呈现给用户。这将是整个项目成败的关键。因此，产品经理的设计是产品经理最核心、最重要的能力之一。

本文将尝试着从51.com项目的实践，来尝试着为大家分析下产品经理的设计力是什么。

Web 2.0兴起未久，各个公司各有相关的实践经验，产品项目的流程也就各不相同。在51.com，我们的产品设计流程如下图所示，产品团队中的各个角色按自身的不同职责在产品经理的指挥下各自展开。



需求分析	产品经理(Product Manager)	细分市场、功能特征、信息架构
	数据分析(Business Intelligence)	采集、整理、挖掘、展示数据
	用户研究(User Research)	用户调研、竞品分析
	交互设计(Interaction Design)	使用流程、人机交互
产品呈现	视觉设计(Visual Design)	视觉界面
	内容设计(Content Design)	文字、文案与内容设计
	界面制作(User Interface Code)	切割页面

需求分析能力

需求分析决定了用户诉求、细分市场、产品受众、功能特征，并最终输出产品的信息架构与业务流程。相较于产品呈现而言，需求分析更能体现一个产品经理的判断力。

在51.com的产品实践中，我们非常重视这一阶段的工作。可以说，项目的成败，70%是由这一阶段的工作决定了。

我们不主张在竞争对手分析身上下太多的功夫。因为，我们认为，与其关注竞争对手，不如去关注用户。这也造成了51在国内SNS网站中，一直是风格独特、独树一帜的。所有的产品经理在入职时，都会在客服部门体验一周，直接去面对用户的投诉或建议。在做需求调研时，我们非常强调获得第一手的资料，以便保持和用户一致的“同理心”。在很多互联网公司，产品经理更多是坐在办公司里天天面对着电脑屏幕。而在51，公司更多地鼓励产品经理走出去，直接去面对用户。51每个产品经理每周都会至少去网吧一次，亲身体会用户的上网环境，实地去研究用户的操作方式。我们有一句口号：“我们不是典型用户”。并且，这句话被打印成标语贴在会议室的墙上。当团队中对产品的设计方案争执不下的时候，产品经理会带着疑问向用户去求证，而不是简单地凭以Leader的感觉来决定设计方案。

随着网站的发展，产品团队也在不断地扩大。不断有来自业内顶尖的公司人才加入51产品团队。来自百度的David Lee很有感慨地说道，在百度，很少有人像51的产品经理这样亲密的和用户去接触。百度的需求分析，更多的是依赖数据。

当然，光是靠这样的方式与用户交流还是远远不够的。与用户交流，不管是采用一对一访谈、现场观察、焦点小组、还是问卷调查，都只是很

小量的定性研究。定性研究的目的并不是定下最终的结论，而是在于最大程度地了解用户特征，提出用户需求的假设。定性研究之后，就需要定量的数据研究的介入。定量的研究可以检验在定性研究阶段提出的各种假设，去伪存真，获得最终的结论。当然，有时候定量的数据研究会推倒所有的假设。于是，我们只能回过头去继续去做用户调研以提出新的假设。这样的循环迭代继续几次之后，就能得出一个比较接近的用户需求。

在得出用户需求之后，我们便可以很容易地确定细分市场和产品受众。并在此基础上进行功能特征、信息架构与业务流程的设计。在这方面，我们严格遵循“少即是多”的设计信条。所有功能点，紧紧围绕用户的核心需求进行展开。产品初次上线时，不追求大而全，而是先实现一个最小的可用模型尽快上线，上线后再根据用户的反馈来决定是继续遵循着原有策略进行完善还是调整方向。这样，就能保证出错成本的最小化和用户体验的最大化。

在需求框架都得以确定后，产品经理会输出一套文档，包括需求文档、使用流程图和低保真模型。此后，就进入了产品呈现的设计阶段。

产品呈现能力

产品呈现的设计是在既定的需求框架下，为产品最终形态呈现而进行的视觉界面与交互设计。

在产品呈现的设计上，具体的设计是由视觉设计师和交互设计师来承担，产品经理更多是扮演了指导与审核的角色以保证产品的最终呈现符合需求的。

在07年以前，设觉设计师和交互设计师并没有参与到需求分析的阶段中。他们只是在需求分析完成之后，才被产品经理告知相应的项目需求。这样，在实践中便产生了很多沟通上的问题。仅凭文档，他们很难理解产品经理的意图。即使后来我们要求双方一定要在文档之外加强口头沟通，但也很难达到理想的效果。

07年以后，我们在项目流程上进行了比较大的变革。设计师被要求参与到需求分析的阶段中。他们甚至有时候会一起随产品参加用户座谈会。产品经理也更多地参与到产品呈现的设计阶段中去。我们意识到，一个产品经理，如果没有足够的视觉与交互设计能力，也很难和设计师达成良好地配合。自从进行了流程的变革，设计师能更好地理解产品经理的意图、用户的需求。产品经理也大大地提升了设计能力。

在没有进行流程变革之前，设计师进行设计主要是根据自我的审美观来进行设计。而进行流程的变革之后，设计师更多地去倾听用户的意见、反馈。请看下面两张界面效果图。上图是51开放平台新上线时候的展示效果，下图是经过改版后的效果。我们刚开始做开放平台时，没有太多的经验，更多地参照了facebook的产品设计。因此，呈现出来的界面也跟facebook很像。当产品上线后，我们发现很多用户在抱怨这种展示效果使用起来很不方便。在接到用户反馈后，我们的产品经理立即联合设计师一起做了一轮用户调研，并在此基础上推出了新版的产品界面。新版的界面没有像facebook那样采用列表和属性菜单进行展示，而是采用类似网址导航站那样的直接展示。上线后，用户好评如潮，而应用程序的添加量也迅速上升了了30%！

产品设计中的科学与艺术

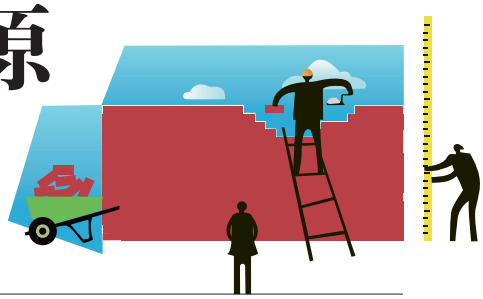
说到设计或者说策划，人们总是会想起那些天马行空的艺术家甚至夸夸其谈的雄辩家形象。但实际上，在产品设计中，确实是存在着艺术的一面，也就是所谓的“凭感觉”；但更存在着科学的一面。也就是说，没有感觉不行，仅靠感觉更不行。

产品设计的艺术，取决于产品经理自身的嗅觉、经验、阅历甚至运气。而产品设计的科学，则需要产品经理严格遵循设计流程、熟练掌握各种技能、方法（包括进行用户访谈、现场观察、组织焦点小组、设计问卷调查、采集用户数据、分析数据、用户角色分析、细分市场、信息架构、易用性、用户心理研究等）。

产品设计中科学的一面保证了能把产品做得对（Do the right things）；而艺术的一面则用来追求把产品做得好(Do things right)。

一个优秀的产品经理，不仅要具备宏观的需求分析能力，还要擅长微观的产品呈现；既要拥有艺术的产品设计感觉，更要掌握科学合理的设计工具和技巧。■

傲游的设计思想之源



■记者 / 付江

互联网和软件公司产品经理这个职位很特殊，在中国还很缺乏，最容易做也最难做。容易的是不需要非常高深的专业学历和很丰富的经验可能也能承担下来，而某些有很高的学历或开发经历的人，却未必能做好。在傲游，如何做产品经理？傲游最重视产品经理的哪些素质？傲游的设计思想来源何处？带着这些问题，记者采访了傲游公司CEO陈明杰（Jeff）。

产品设计要有前瞻性

傲游要求产品经理不仅要对用户需求有深刻理解，更要注重对用户需求的提炼，产品设计要有一定前瞻性。从某种程度上，产品经理就是代表用户的发言人，而且要时常把自己当作挑剔的用户，笨的用户，懒的用户，产品的设计要符合用户的习惯，从一些细节的点来赢得用户，让用户在感觉、触觉上都用得很爽，如鼠标可以少移动，能迅速点到，按钮放左边还是右边哪个更接近用户习惯，输入框是否采取智能形式。傲游在这方面做了很多尝试，例如在傲游浏览器右上角增加截屏按钮，在右下角设置缩放页面的快捷，在左上角设置多重搜索等等，都获得了用户较好的反馈。

由于每天都在和各种各样的与自己的产品相关的需求打交道。只有对这些需求进行有效的提炼，从深层次分析用户提需求的原因，才能设计出

傲游已成为基于IE内核的重要浏览器产品

真正有价值的产品模型，否则只是徒增开发人员的负担而已。傲游要求，对于用户的需求，不是说用户需要功能A，就给他简单的提供功能A。更为重要的是，能够从更深层次的，更具有远见性的，了解用户为什么提出这样的需求，这种需求是否代表广泛性，用户未来还可能提出什么需求。这样在产品设计上既可以解决用户A的需求，又解决了用户将来可能提出的需求B。Jeff也总是要求傲游的产品经理们对设计不仅要适应目前的需求，更要具有前瞻性，他自己也在时刻考虑如何让傲游在未来几年都在市场上保持领先优势。

关注最核心功能 订优先级次序最重要

产品功能最难的是订优先级和先后次序，产品经理要懂得自己产品的核心竞争力在哪里，哪些功能是用户最关注和最核心的，将核心竞争力发挥到极致，将最重要的功能优先实现。在这方面，傲游也走过一段“弯路”。举个例子，原来傲游的产品经理设计过一个类似iGoogle的个性化主页功能，最初大家讨论的时候都觉得这个方向很好，一部分比较高端的用户也确实有这个需求。但在这个功能实际开发过程中，发现要做好这个功能还是需要投入很大的精力的，而投入过大的话就会直接影响到浏览器其他功能的开发进度。在内部充分分析个性化定制的开发难度和投入产出比后，再考虑到公司当前的状况、人员配比和手头任务的紧要程度，最终在这个功能测试版即将上线的时候被砍掉了。

通过这件事，Jeff和傲游的产品团队更深化了三点认识：(1)对于一款在广大用户群中拥有较好口碑的产品，每添加一项功能都要慎重考虑，分清孰轻孰重，分清一个功能主要是满足10%用户需求还是剩下90%的用户的需求，在给10%的用户带来好感的同时是否也会给90%的用户带来迷惑。(2)功能好不好不仅要满足用户的需求，并且要注意在适当的时间推出适当的

功能。(3)如果产品团队分不清方向，让兄弟们陪着干，结果发现方向错误是非常浪费和挫伤团队士气的。面对多方面的需求，产品经理要懂得抉择，学会舍弃，才能最大程度上调动公司资源，解决用户最核心的问题。

傲游的插件系统是Jeff个人比较满意的一个功能，不仅因为这个功能的整个系统框架是他自己设计的，大部分代码也是Jeff亲手编写。在设计插件系统的时候，从一开始就采用了跟IE不一样的模式，目的就是要建立一个更广阔的平台，支持插件的种类更多，让更多的人可以参与进来。“现在傲游的插件数量是所有基于IE内核开发的浏览器中最多的，”Jeff自豪的说道。在采访的全过程中，技术背景出身的Jeff多次流露出对编程的爱好，“在傲游最初的时候，很多功能都是我亲手开发的，虽然现在直接写代码的机会少了，但还是一直对技术研究性的内容很感兴趣。其实并不是我不想写代码，主要是因为没有时间，再加上如果我突然写一段代码插到程序中，开发组的同事反而不好接受，所以才少写一些。但公司的技术研讨会我还是会尽量参加的。”Jeff对技术的热爱由此可见一斑。

产品设计思路从哪来

傲游浏览器设计思路主要来源于四个方面：个人体验；用户处收集；学习同类产品长处和创新。Jeff坦言，最初做MyIE2的时候，设计思路更多的来自于个人体验，因为自己和合作伙伴都是网虫，在使用浏览器的过程中，他们自己发现了MyIE2中的很多问题，针对一些突出问题，Jeff会自己做出相应的功能改进。

对于从用户处收集到的需求，信息往往零散且杂乱，从某种角度看，会呈现一定的片面性和局限性，这就需要产品经理做进一步分析和提炼。例如，有的用户表示，浏览器越简单越好，但如果真的提供一个只能上网，啥功能也没有的浏览器，用的人也未必见得多。其实用户在表达越简单越好的时候，不代表他不需要更多的功能，或许他只是怕功能多了以后会影响到浏览器使用的流畅性。因此，对于产品经理更重要的是分析能力，对于用户的反馈能在各方面因素中找到一个最好的切合点。

对于同类产品间的取长补短，Jeff毫不避言“学习同类产品的长处丰富傲游也是必然的。浏览器的开发思路开始趋向一致，多页面、多进程等等技术大家都在跟进，好的功能主流浏览器都会做。互相取长补短，才会促进整个产业更快的向前发展，用户也能得到更具用户体验的产品。”但Jeff同时强调，“在这方面傲游的原则是，对于同类产品已经放出来的功能，傲游绝不会简单的将类似的功能直接Copy过来，一定会在原来的基础上有进一步的发展，为用户提供具有更好体验效果的产品。”

除了以上三种情况，傲游长期以来也一直保持着对产品创新力的高度重视和“饥渴”状态，积极加强在产品商业模式和技术上的创新。在商业上，傲游是首个采用捐助软件(Donate ware)形式的产品；在技术上，傲游是在所有IE内核浏览器里第一个支持IE插件的。早在MyIE2时期，傲游就率先实现了支持Google工具栏，FlashGet工具栏等各种IE插件，也支持了迅雷等各种下载工具。傲游作为第一个有比较完善插件系统的IE内核浏览器，从文档到示例，都有很丰富的资源。不仅如此，傲游还是首个支持多内核的浏览器（傲游1.x，支持IE和Gecko的内核）。并在世界范围内率先实现了支持帐号系统和内嵌在线收藏系统的功能。所有这一切，保证了傲游浏览器在网民中一直有较好的口碑和普及率。

傲游目前正在积极拓展海外市场，Jeff表示，由于国内外用户的使用习惯不同，在产品的设计上也会有一些不同的侧重点。比如海外用户使用互联网的时间比中国网民更长，他们更偏爱个性化、定制化的功能和内容展示，因此海外用户可能会更容易接受iGoogle这类的产品。而国内目前很多新网民接触互联网的时间还不长，他们或许更急需在产品中增加易用性和能够迅速解决他们上网需求的功能。这更多是出于互联网使用时间和用户习惯，以及思路的不同。傲游也会尽量在个性化和易用性方面做到一个最佳的结合，这一直也是傲游努力的方向。

未来更重视设计

Jeff认为，目前中国软件和互联网公司对产品的重视程度还是不够，尤其在很多创造性和细节上，而好的产品设计是公司的灵魂。如果不注重产品用户体验和实用性的提升，过多的做市场活动、推广，甚至是捆绑销售，这些都是浮躁的体现。或许短期能获得一定利益，但如果产品做得不过硬，做过多的推广只能让用户得到更多的失望，而用户口碑一旦坏掉后，要想再拉回来，就非常难了。

中国不缺乏经验丰富的开发人员，但优秀的产品人才还是远远不够的，资深的开发者往往由于过分关注技术，而忽略了产品本身能给用户带来的价值。对技术自信没有错，但若处理不好产品设计的问题，这样的产品也很难成功。对于开发人员创业，Jeff建言，业界未来对产品的设计会越来越重视，开发人员要想成功，或者自己多花一些时间研究产品的理论和知识，或者找一个好的产品团队做拍档，这样好的技术才能更好的发挥。

最后，Jeff透露，大家所期待的傲游3正在有序的开发过程中，和傲游2相比将会采用全新的设计和架构，在用户体验上仍将延续过去的一致性。而他自己也参与了傲游3架构和框架的设计。如果顺利的话，明年大家就会用到全新的傲游浏览器了。■

普元：执行力打造企业级平台产品

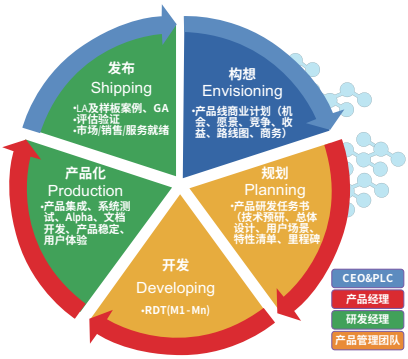
在普元，产品经理既是政治家，又是企业家，还是总经理。——焦烈焱

■记者 / 田硕

普元产品研发观

今天的普元，产品定位为“SOA中间件”，有两条产品线：EOS（针对SOA中的服务）、BPS（针对SOA中的流程），每个产品线有一位产品经理。这两位产品经理，负责了其产品的规划、开发和产品化几个重要的环节，他们为最终产品进入市场的成败负责。

从严格意义上说，整个普元的高级管理人员团队都在从事产品相关的工作。具体说，普元产品研发的过程源自IPD流程，是一个基于市场的并行产品研发体系，产品研发过程中会有一个产品管理的团队，包括市场、销售、研发、服务、验收、用户体验和技术架构等角色。这个过程包括了产品构想、规划、开发、产品化和发布五个阶段：



● 构想

构想阶段产品经理需要组织资源完成产品的商业计划，包括市场定位、竞争对手分析、产品收益评估和风险等，提交PLC（产品领导委员会），获得通过后进入下一阶段。


普元首席架构师焦烈焱也是其产品团队中的重要角色

● 规划

规划阶段的Owner是产品经理，这个阶段产品管理团队会制定产品研发任务书，除了对构想阶段内容继续细化外，还有制定里程碑和资源计划，典型用户场景和大粒度的需求（不是详细需求，例如此时需求会定义“支持业务模块动态部署”，具体如何动态部署，是开发阶段做的事情了）。

● 开发

开发阶段会成立PDT（产品开发团队），在这个阶段每个产品经理的工作重点会根据自己的经历、背景做出不同的选择，但是进度、质量是一定要关注的，在这个阶段产品经理也需要做未来客户的沟通、市场策略的定义，为未来的产品找到试用的客户。

● 产品化

平台产品的产品化过程比较耗时耗力，例如在目前普元的大版本研发，LA（有限发布版）和GA（正式发布版）之间至少有半年以上的时间，同时，市场宣传也同步启动。

● 发布

在产品正式发布前，产品经理需要确保服务保障就绪、营销策略和工具制定，以及制定的销售策略和销售工具能够被理解与执行（白皮书、市场准备、销售培训、案例），内部资源的到位（服务、支持）。

从这个结构中可以看出，普元的产品经理，在实际的职责上更偏向于确保产品能够及时交付，而按时交付不仅仅是按时开发完成。其实这也代表了企业级产品与一般软件产品的不同：更注重产品的适用度。

企业级平台产品特点

焦烈焱认为：“本质上说，产品经理之间没有什么差别。不同点内部来自各个公司不同的管理文化，外部来自面向的客户群。”当然，由于每个产品经理的经历不同自身上会有一些取舍，例如产品经理的业务背景强，他本人可能会在这方面更富有权威。但是，产品管理是一个团队，产品经

理是这个团队的Leader（不仅仅是Manager），他需要发挥团队的优势确保产品的成功。

企业级平台软件产品经理如果和其他软件相比，共同点是都必须关注产品的商业模式，比较大的不同是在目标客户的选择，因为平台软件的受众比较复杂，所以在市场细分、产品预研和规划、有效收集客户需求、支持服务体系建设、市场推广等方面与应用和互联网的产品经理有较大不同。

既然产品经理是产品管理团队的Leader，而不仅仅是Manager，那么对于这个职位的要求，也相对更高。然而，如果说对产品经理的要求是比较全面的，不如说对产品管理团队的要求更全面。“我们的产品经理需要能够调动内外部的资源，尤其是外部资源。”焦烈焱提道。

此外，普元在产品上会有很多需求来自客户的反馈。但是对于多数平台厂商的产品经理来说，不是需求太少了，而是太多了。焦烈焱表示：“我们必须瞄准行业或者某个典型意义的客户，展开需求。在产品使用过程中总会有一些新的需求，对普元的产品经理来说，需求管理是很重要的，我们一般每年都要做大的版本规划，每半年都会修正一下，期间产品经理的规划依据是普元销售、市场、服务等各个环节来的需求，而且对重要客户群普元的产品经理都会亲自到客户那里去调研，还会考虑到竞争优势，最终产品经理根据这些将产品需求丰富到各个版本中。回过头来看，很多需求并不是都适合在产品中体现，这时会对需求重新做定位，对于产品经理来说，更加关注的是这些需求所属的问题域，这一类行业用户中共性的要求。”

关键因素：执行力

在了解了企业级平台产品的特点之后，需要确保的就是产品化的具体过程。这一点从普元的产品经理责任流程就可以看出。尤其是在交付问题上。

焦烈焱认为，能否按时交付是一个大的系统工程：“很早以前我对按时交付的理解仅仅是如何按时的开发完成，现在的感受不一样了，按时开发完成并不意味着按时交付。”这是因为，除了开发完成的交付之外，还需要做很多额外的工作，包括与合作伙伴完成样板案例，让产品通过一定的评估与验证，从而达到市场、销售、服务等工作的就绪状态。为了保证这一过程的顺利执行，需要良好的执行力。

“普元历史上主要的版本是3、5、6这三个系列，我们没有4版本（据说做4版本的都失败了）。目前的产品管理体制也是在这么多版本经验的积累上逐步形成的。”焦烈焱介绍，普元的产品管理大致会分为三个阶段：

- 早期自发的产品需求管理：产品的定位、需求都是由研发根据用户的反馈以及技术发展的需要确定，但是后来会发现研发的产品和用户的使用场景有出入，用户的满意度不高；
- 中期，面向内部用户的产品管理：为了改变早期的现象，保证研发出的产品确实是用户需要的，我们采取了一些改进，普元内部有两个技术团队：研发和专业服务，前者负责产品的开发/测试/架构/预研，后者负责售前/咨询/技术支持/培训等工作。这一阶段，专业服务团队会作为产品的内部用户，也就是一个虚拟的用户，对产品研发提出具体的要求。但是后来也发现这种结构过于关注具体特性，对产品市场的关注度不够；
- 目前，以用户为导向全流程的产品管理：目前的产品经理制是前BEA中国总经理沈惠中加盟普元后确定的，产品管理从过去产品开发的管理发展为全流程的产品管理，产品经理对产品的成败全权负责，普元的两位产品经理刘尔洪和程朝晖都具有多年的市场、技术背景。应该说普元现在的产品管理更加以用户为导向，这些都需要建立在良好执行力的基础上。

随着SOA相关标准的成熟，开放性和标准化作为6系列版本的主要特性。恰在这个时候，普元在世界软件工程大会上遇到了IBM Rational的CTO，他发现普元产品很多思路与SCA相似，普元基于这些思路大量的案例会对SCA的发展很重要，而普元也认识到SCA/SDO可以让产品理念更开放，所以当普元正式提出加入标准制定时，他只问了一个问题“你们是认真的吗？”。

后面的过程比较顺利了，普元甚至早于SUN加入了OSOA组织，目前这一标准已经提交到OASIS组织，发布了SCA1.0、SDO2.1版本，在IBM、Oracle、Tibco等产品中都得到了广泛的支持。

相信常常参加国外会议的中国企业，在面临那句“你们是认真的吗？”时，都会或多或少的犹豫，而普元正是在这样强大的执行力和行动力下，成为了行业的领先者。

前进中的探索

当然，任何一个产品，都有其客观存在的问题。而这些问题，也是每一个企业从创业阶段就开始不断面对的。普元产品研发最大的问题还是目标客户群的定位，因为普元是一个通用型的平台，这样的平台如何适应不同客户的诉求，是一个巨大的挑战。产品经理职责中最为关键和困难的就是搞清楚符合公司定位和市场所需的产品定义及其产品规格，也就是说做对事情是首要的。

这一问题焦烈焱直言，普元正在努力通过IPD的一些思想更好地解决。这种定位，不能仅仅从技术角度看，更多是从用户业务角度出发。之所以普元对这类问题感触比较深，更多的原因是因为通用平台性软件的特殊性，但是平台性软件也有客户的细分，更需要解决客户最终的业务问题，例如WebLogic的成功是因为解决了用户把业务投入到互联网应用中，而不是因为用户有了B/S这样的技术架构需求。■

市场与 Windows 的双向选择

■ 记者 / 江汉

很多程序员都有一个梦想，就是开发一个让全世界数亿人都使用的软件。这一点来说Dennis Flanagan确实让人羡慕，因为从Windows 98时代，他就开始负责Windows产品线的研发和市场推广工作。可以说Dennis见证了微软的成功，也见证了这家由一个产品风靡而建立软件帝国的历程。

不是秘密的秘密

若是让我们枚举微软所创造的历史，恐怕足够说上整整一本书。从微软的企业文化到微软的经典里程碑开发流程，再到其领导人比尔•盖茨的传记。微软似乎早已没有秘密了。

然而，如果非让我们从微软众多功绩中挑选一两项作为其成功的要素，那么不得不承认微软在构建大规模客户端产品的能力上独树一帜。这种大规模，除了包括客户端软件的代码规模十分庞大外，还包括整个微软技术体系的庞大，更重要的是，数量巨大的微软客户端用户，更是让人敬畏，这是今天极少有企业能够做到的。

尽管一直到今天仍然有不少人在非议微软软件产品的质量，喋喋不休地纠缠微软产品的安全性问题，但事实上，除了微软，已经没有任何第二个公司能够做到自己的产品被这样庞大的用户深度使用，仍然能够确保其产品的代码质量和有效。主流软件工程的思想尽管一直没有在微软体系当中得到大力的宣传，但微软却在主流之外创造奇迹。


从Windows98时代就开始从事产品设计的Dennis Flanagan

Windows产品总经理Dennis认为：“开发任何大规模产品，尤其是像Windows那么复杂的系列产品，不仅会涉及许多软件系统，还有更多硬件系统。这需要微软有一系列基本思路，一个结构非常完善的工艺流程，使得整个团队可以按部就班地协作。”

当然，这不仅包括产品的开发，而是从产品在总体规划、功能的设计和添加、开发、测试以及市场推广等多个环节的协作。仅仅在Windows 7研发团队，有三个不同的小组在协同工作，他们主要负责开发、测试等工作。工程的进度已经确定，何时能交付却并不是那么容易控制。其实，Windows成功推向市场，主导权不在编码人员编写代码的时候，也不在于管理人员协调资源的时候，而是更高层面上的工程理念控制。比如今天我们要在Windows 7上面添加一些额外功能的话，就必须在事先制定非常完善的工艺流程。

正因为有效的工艺流程，微软机器拥有了将产品推向市场的动力引擎。

特例无法复制

然而，仅仅凭借流程和工艺，仍然无法把软件变成商品。Windows的成功，其实可以看成一个特例。这也是自微软成功二十多年来，一直没有后来的软件公司能复制这一模式的原因。

如果我们回顾一下20世纪80年代的市场，我们就可以很容易地得出这个结论。那时的人机交互最常见的就是Unix式的命令行模式。这对于科学家来说当然不是难事，但对那些普通的家庭主妇，恐怕光是要记住各种各

样的复杂命令，就已经足够让她们头疼了。尽管那时家庭主妇还没有今天这样对PC的需求，但毫无疑问的是，市场有了一个缝隙，它被微软抓住了。

接下来的故事，想必每一个开发者都知道。Windows的横空出世一举改变了人们使用计算机的方式，这也很快把整个计算机市场升级到全球每一个家庭的桌面上。常常有历史学家假设，如果当时做这件事的人不是比尔•盖茨，那么会不会有另外一个人来完成这个奇迹？基于这种假设的结论通常都是肯定的，计算机本身的发展，对于历史提出了需求，也造就了微软。

回顾这段封尘的历史，我们已经开始对历史的市场进行分析和反思了。**一个具备新理念的产品在其开拓市场阶段，并不需要太多的市场行为来宣传和推广，因为新产品的理念其本身就是市场推广中最重要的部分。**这也是Windows 95成功的原因。但微软并没有因为这个产品的成功而停下开疆拓土的脚步，而是深入挖掘了成功之后的价值。

建立市场生态

是的，就如同Dennis所说的那样，Windows是如此庞大的一个技术体系，以至于每一次对它的升级和改造，都面临了前所未有的难题。操作系统平台需要得到有效的支撑，才能让这个产品在市场中长盛不衰。各种各样其他的软件系统如何在这一平台上流畅运行，操作系统如何与底层系统进行交互，诸如此类问题不是微软一家厂商能够把握的。

合作伙伴关键的建立，成为那时候微软需要度过的难关。此时的微软已经不再是开发Windows 95时代的方式了。从Windows 2000（Windows NT 5.0）开始，越来越多的硬件和系统厂商加入到了微软合作伙伴阵营，这让他们可以开发出直接使用在这个操作系统平台上的软硬件系统，从而确保他们能够有非常完整的兼容性。这种情况一直持续到今天，我们甚至可以在很多外部设备上看到Windows产品的标示，这也代表着以微软为中心的生态系统正在逐步建立起来。

这项工作在Windows XP时代几乎达到了极致。随着微软与合作伙伴及客户的合作持续紧密，微软开始接受越来越多的挑战。由于近乎已经垄断了市场，如果今天你作为开发商不支持微软的技术体系，那么你的产品就很难在市场上立足。

与此同时，微软自身也并没有停下对整个平台体系的构建。以Windows为核心，各种各样的产品被同时开发出来，这当中不仅仅包括针对开发人员所使用的开发工具，还包括各种企业级服务器产品和消费类用户所使用的日常软件。Office、IE等一批Windows平台上的核心应用产品将用户更进一步地绑定在Windows平台周围，微软的技术生态圈已经趋向完成。

新策略：深耕密植

完成了生态体系的建设，就可以说微软已经可以不再需要进步了吗？从我们看到的历史，可以认为微软前进的脚步从未停止过。Dennis在接受采访的时候提到的一个观点更让我们看到了后Windows时代，微软策略的变化：“我们要保证客户在这方面投入的每笔钱都是值得的。”

这是一个宣言，一种态度。以Windows 7来说，“我们打算设计Windows 7的时候，已经在计划阶段跟客户、硬件厂商、合作伙伴会商，跟他们说出我们的详细计划。这可以确保当他们见到我们推出Windows 7的时候不会感到吃惊，因为他们已经知道我们会做出什么样的产品。整个产品的开发过程当中，我们都在不断整合这些人的反馈和需求，等到最终产品面试的时候，他们会发现其需求已经反映在了产品当中。”Dennis提道。

将合作伙伴的利益绑在微软战车上，让微软的推广不再是一个独立的行为，合作伙伴的产品和推广，同样代表着微软技术的成功。持续强化产品生态系统的能量，让整个市场形成一个整体，这样才能真正意义上确保微软的技术始终处于核心位置，这是作为开发人员原来不曾想象过的战略。

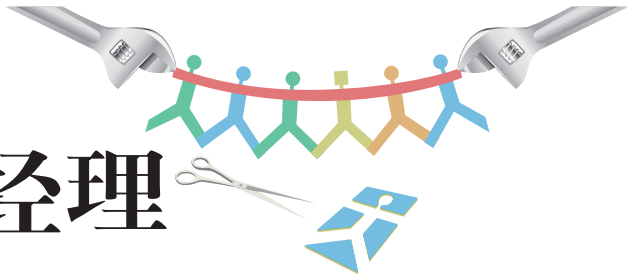
当下，全球有超过600万的微软技术开发人员，这些人员在对微软技术体系的学习投入，似乎都有不菲的回报。其实，这本身就是微软战略的一部分。大量的人才，帮助了这个市场逐渐走向成熟，随之而来的是技术微软技术的企业能够获得充足的人力资源。在解决了许多传统企业在产品推广上的问题之后，试问微软怎么走向失败？

从某种程度上来说，基于微软技术的开发者都是这个生态系统当中重要的一环。仅仅是Windows平台上的共享软件，就已经造就了一个巨大的用户群体，更不用说那些将微软的技术运行在各种商业业务当中的开发商了。

结束语

即使今天的微软已经是一个庞大的软件帝国了，但它对待产品的态度，仍然坚持着小学生的态度，用严谨的方法一步一步深入其产品推广。我们今天已经再难重现当年微软设计产品的种种细节，从大的市场层面上，我们却看到了微软是如何将自己与用户、合作伙伴结成一个整体的。尽管在这样的局面下，用户和产品提供商之间仍然会存在这样那样的问题，但这种局面已经让双方都无法退出。

当年比尔•盖茨曾经说过这样一句话：“任何一家公司，距离倒闭只有18个月。”但真正将产品与市场结合起来进行双向选择的企业，其坚持的时间要远远长过这个判断，Windows 7的出现，能否让微软持续撰写奇迹的历史，让我们拭目以待。■



产品经理就是总经理

■ 记者 / 韩纲

2006年，傅盛从雅虎出来，重新加入了老东家周鸿祎的奇虎公司。此时的奇虎公司创立时间还不长，却已经有了一张未来的蓝图。在当时3721担任了三年产品经理的傅盛，带着三个人开始了360安全卫士的产品之路。现在，360安全卫士已经成为了家喻户晓、耳熟能详的安全软件了，而傅盛却改变了自己的人生轨迹，加盟经纬中国成为了一名投资人。

今天的傅盛已经是一名投资人了，所做的工作虽然有很大变化，但他仍然非常关注产品，以及产品团队。在他看来，一个值得投资的项目，一定有一个值得投资的产品经理。而被投资的产品，必须有自己独特的亮点和明确的思路。这是运营一个软件产品所必须具备的原则。

今天的傅盛扮演者产品经理与投资人的双重角色

产品经理要把自己看成总经理

许多公司都面临着同样的一个问题，为什么产品经理这么难招？傅盛说：“产品经理既要有能够把握产品和市场方向的能力，又能够静下心来做很多耐心细致的工作，甚至于包括设计产品的细节。这种要求对于一般的技术人员来说确实是太高了。”这也是为什么在很多公司里，总经理就是产品经理。

但是，在很多的公司，产品经理其实是一个定义很模糊的角色：作为一个产品的负责人，这个角色通常承担着大量压力和责任，却几乎没有权力可言。由于产品经理本身并不是一个管理职位，这就对产品经理提出了非常高的要求：能够想方设法协调各种公司部门的资源，为之己用；如果产品在市场上出了问题，必须能自己肩负责任，承担风险；所有工作中最琐碎的部分，基本上都是产品经理一手完成，事无巨细。用傅盛的话说，这是一个生存环境很艰苦的职业。

如果用电影导演这样的角色来做对比的话，就可以发现与产品经理的工作很类似。在面对灯光、摄像、场景、剧务、场记、后勤、演员、音乐、后期等一大堆细琐的工作面前，导演必须坚定不移的努力完成自己的作品。因为电影拍砸了，人们只会认为导演无能，没有人会去责怪其他任何一个角色。由于权力并非绝对存在，即使是大导演，在面临那些不能开罪的人面前（比如制片人、投资人以及大牌演员等），也只有耐着性子慢慢打磨。

产品经理关键词：积极心态，关注细节，善于沟通

最开始的傅盛也和今天很多产品经理一样，一心只想做大事，对于很多细节的工作显得三心二意。加上他以前曾经开发过不少MIS系统，其中还有些是直接给国家各大部委开发的项目，面对一个产品页面上的小按钮，傅盛怎么也提不起兴趣来耐心揣摩。尤其是在最开始担任3721产品经理的时候，“上网助手”产品的界面几乎简单到了没什么事情可做的程度。

在初任产品经理的那段日子，什么事情几乎都得自己做。团队里一共没有几条枪，产品设计出来，连测试都觉得很难看，索性干脆罢工了。这种时候，积极主动的心态在做产品的过程中起了非常重要的作用。按照通常的情况，多半产品经理都会认为测试不给自己面子，但傅盛觉得，这是

作为产品经理不必可少的思维方式。

从心态上来说，傅盛认为有两种人绝对不适合做产品经理。第一类是那些自以为是，刚愎自用的人。这类人通常独断独行，很难与人沟通。要么完全听不进任何意见和建议，要么阳奉阴违。另外一类则是那些耳根子太软的人，在面对关键问题上优柔寡断，甚至在某些争执发生的时候采用骑墙策略，无法对产品本身有实质性地推进。有意识地培养心态，是产品经理的入门课程。在遇到批评和意见的时候，第一要诀就是放弃反驳，即使别人说得不对，也应该至少静下心来听听，放下面子，才能真正做好产品经理。

关注细节是一个产品经理必备的能力。现在回想起来，当初觉得3721上网助手界面无事可作其实也是自己对细节不够关注的一个表现，傅盛回忆道。现在再看看浏览器的地址栏，里面有太多的产品设计可作，Firefox 3也正是在这里找到了突破点，一鸣惊人的。现在，面对产品界面、网站、宣传稿等，傅盛一眼就能看出里面非常细微的差别和不足，下属常常惊叹他的“眼尖”，也很头疼和他一起设计产品的界面或是琢磨如何宣传给用户，因为总会被他尖锐的指出诸多不足。

“其实我生活上是一个很马虎的人，平时都不怎么修边幅，以前也从来没觉得自己能在细节上这么关注。之所以现在能够到达这个程度，没别的秘诀，就是熟能生巧”傅盛介绍到，为了能让自己对产品界面等细节有更好的感觉，他不断的使用各类互联网产品，尤其是客户端产品。每个产品发布新版的时候，自己也是第一时间下载、试用。日积月累，细节能力就出来了。有句话叫“读书破万卷，下笔如有神”“熟读唐诗三百首，不会做诗也会吟”，讲的就是这个道理。

由于产品经理的沟通工作横跨开发、测试、运营、市场等多个环节，因此有效沟通是产品经理的核心能力。“产品设计工作中，80%的问题都是沟通问题”，傅盛这么定义。很多时候，产品从最初的设想到最后的发布，差别巨大，很重要的原因就是没有沟通清楚，一点一点的差异的累计，导致了最后的巨大变形。“我们看很多大片，最后被观众骂的一塌糊涂，我就总在想，难道这些导演们不知道自己拍的片子烂吗？很明显，他们自己也知道。但可以肯定的是，最初，在他们脑海里的片子一定是一部美轮美奂的样子。但是，就是由于和各方面资源的协调、沟通的误差，导致最后快成型的时候，完全不是自己想象的样子了”一直喜欢用电影做类比的傅盛得出这样的结论。为了能够沟通清楚，傅盛逐渐养成了“逼问”的习惯：即在表达完一个观点之后，要对方再表述一遍，还要把观点拆散从其它角度追问对方的理解程度，才算把一个事情沟通完毕。

具备猎狗一样的嗅觉，准确找到用户的甜点

当然，要具备产品观并不是一件容易事。“每一个产品都应该有一个明确的突破点，今天看，几乎没有任何产品，一上线就开始搭建平台而取得成功的。尤其是面向互联网的客户端应用产品，如果做成大杂烩，几乎不可能成功。这方面我们曾经见过中搜出品的网络猪，把浏览器、搜索、下载、聊天、邮件等所有的功能都集成在一起，摆在用户面前的必然是一个毫无特色的产品。”傅盛提道。

把握用户的需求来规划产品，尽管是一个目标，但绝不是被用户牵着鼻子走。更何况，踊跃提出“宝贵意见”的用户，往往都是一个小众，而产品规划的核心要点，是需要通过机制将大多数用户放在最重要的位置上。这一点几乎是很多互联网上曾经流行一时的小产品的通病，这些产品往往在进行产品功能规划上遇到了瓶颈，正是因为他们自以为是地认为把握了用户的心态，却忽略了产品本身的内涵。

尤其是那些小有口碑的产品，其产品团队很难自我否定，最终满足于单纯的产品功能，忽略了产品背后提供服务的那部分真正价值。甚至有些团队醉心于自身的技术，一味扩大产品线，无法在一个产品上做透，做扎实。“技术永远不是最重要的，只有和商业有效结合，才能把技术用好。”傅盛如是说。“比如iPhone，从技术本身而言，没有一个技术是独创的，但是，把很多实用技术捏和好，找到一个突破点、奇迹便会发生。”

做了投资人后，傅盛对此有更深刻的感受。“如果你把一个很领先的技术放在风险投资人面前，他们的眼里不会像我们一样闪烁着兴奋的光芒，反而，他会非常冷静甚至挑剔的追问，这个技术怎么应用给大众，能够找到怎样的商业模式，还有，这个技术的领头人是不是有足够的商业头脑，有没有足够的团队管理能力，有没有百折不挠的心态。所有这些，才是风险投资人们考虑的重点”。

产品经理的成就感

当然，产品经理是一项很有成就感的事业。产品经理是公司的“无冕之王”。行业内真正成功的产品经理，能成就一个企业。百度的俞军，从一个普通产品经理做起，创造了百度贴吧这个产品，最终成为了百度公司的副总裁。而傅盛，也从一个普通的产品经理，通过拼搏打造了一个全国闻名的360安全卫士，最终变为该业务的总经理，现在又转为知名风险投资公司的副总裁。提道这些，傅盛总是露出自豪的微笑：“有一段时间，我为了弄清楚网络游戏的机制，玩起了魔兽，我的老板当时怕我沉迷于此经常提醒我。我对他说，你放心吧，网游给我带来的成就感，哪有做产品得到的成就感大啊。魔兽在封闭的世界里最多升级到70级，我的产品呢，在开放的世界中边界无限，改变千万人的生活，我因此倍感幸福。”■

大型Web 2.0网站架构纵横谈（一）

■ 文 / 疯狂代码

如今，有许多像海内、开心网这样的大型Web 2.0网站，它们具有高互动性、高负载、高数据流动性等特点。本文不讨论实现语言，而是从架构角度看问题。虽然有些杞人忧天，因为网站发展前期可能不太实用或者不需要考虑太多，但希望能在您的系统遇到瓶颈时提供一些帮助和参考。

首先讨论一下大型网站需要注意和考虑的问题：

A.海量数据处理

小站点的数据量并不大，select和update就可以解决问题，本身负载量也不是很大，最多再加几个索引就可以搞定。大型网站每天的数据量可能上百万，甚至上千万或更多。如果存在设计不好的多对多关系，在前期可能没有任何问题，但是随着用户增长，数据量会以几何级数增加。此时，对于一个表的select和update(还不说多表联合查询)的成本是非常高的。我们需要一个有效的数据处理方案。

B.数据并发处理

2.0网站的CTO都有个尚方宝剑，就是缓存。然而，缓存在高并发处理时也是大问题。在整个应用范围下，缓存是全局共享的。当两个或者多个请求同时对缓存有更新要求时，尽管我们有lock机制，但并不是很灵验，应用程序还是会直接死掉。这个时候，就需要一个好的数据并发处理策略以及缓存策略。

还要注意数据库的死锁问题。也许平时我们感觉不到，但死锁在高并发的情况下出现概率非常高，这时磁盘缓存就是一个大问题。

C.文件存贮问题

对于一些支持文件上传的2.0站点，在庆幸硬盘容量越来越大的时候，我们更应该考虑文件应该如何被存储并且被有效地索引。常见的方案是对文件按照日期和类型进行存贮。但是当文件量是海量数据的情况下，如果一块硬盘存贮了500G琐碎文件，那么维护和使用时，磁盘I/O就是一个巨大的问题，哪怕你的带宽足够，但是你的磁盘也未必响应得过来。如果这个时候还涉及上传，磁盘很容易就over了。

也许用RAID和专用存贮服务器能解决困境，但是还存在异地访问问题。如果服务器在北京，那么在云南或者新疆的访问速度如何解决？如果做分布式，那么文件索引以及架构该如何规划？所以我们不得不承认，文件存贮问题没那么容易对付。

D.数据关系处理

我们可以很容易规划出一个符合第三范式的数据库，里面布满了多对多关系，还能用GUID来替换INDENTIFY COLUMN。但是，在多对多关系充斥的2.0时代，第三范式首先应该被抛弃，因为我们必须有效地把多表联合查询的几率降到最低。

E.数据索引问题

众所周知，要想提高数据库查询效率，索引是最方便、廉价且容易实现的方案。但是，在高update的情况下，update和delete的成本会高得无法想象。笔者遇到过这样的问题：更新一个聚簇索引，需要10分钟完成。这对于Web 2.0站点来说，是不可忍受的。

索引和更新是一对天生的冤家，问题A、D、E是我们在做架构的时候不得不考虑的问题，并且也可能花费时间最多。

F.分布式处理

对于2.0网站，由于其高互动性，CDN实现的效果基本为零；因为内容是实时更新的，我们不能仅对其做常规处理。为了保证各地的访问速度，我们就需要面对一个很大的问题，那就是如何有效实现数据同步和更新，实现各地服务器的实时通信。

G.Ajax利弊分析

成也Ajax，败也Ajax，Ajax成为了主流趋势。我们突然发现基于XMLHTTP的POST和GET是如此容易。客户端get或者post数据到服务器，

服务器接到数据请求之后返回来，这是一个很正常的Ajax请求。但是在对Ajax进行处理时，如果我们使用一个抓包工具，对数据的返回和处理便一目了然。

由于其基于JavaScript，我们可以很清晰地理解它们的加密验证方法。后面有个很典型的例子，通过分析，我们很容易实现QQ空间和QQMail的加密方法，并伪造合理的数据；前者是阉割修改过的md5，后者是阉割修改过的RSA。但是没有关系，我们可以直接拿来用。对于一些计算量大的Ajax请求，我们可以构造发包机，很容易就可以把一个Web服务器干掉。

Ajax可以将用户体验做得很好，经过设置还可以使一些基于WebBrowser的类似组件的HTTP攻击无效，但如果基于HTTP数据包处理方面，Ajax无疑成为一个非常容易构成的攻击点。

很多人对基于Ajax的攻击提出疑问，比如不能跨域、减轻负担。Ajax通过简单的Get和Post进行数据传递，采用HTTP Debuger抓取数据，然后使用如下方案，顺便写个示例的攻击代码。相比传统的网页表单，我们更容易构造一些。其实对于网页表单和Ajax来说，处理和发包过程是一样的，Ajax数据量相对小，速度也快一些。

下面我们用伪代码来实现基于腾讯QQ空间的群发方案：

```
request.CreateUrl(Ajax或者http处理页面);
request.Method=”GetORPost”;
request.refere=”网页来源”;
request.Cookies=”这里附加登录时服务器给予的cookies，如果不采用Ajax，我们设置还可以直接构造cookie过去，绕过登陆”;
request.params=”这里的参数是QQ修改过的md5验证”;
/*对于QQ验证码的处理，QQ验证码识别相对复杂一些，笔者没有能力实现破解，但是还有个另类的处理方法。我们可以让其他人帮我们实现。我们在另外一个Web界面设置是自己的宿主应用程序上要求用户登陆，验证码就用QQ的，大家应该明白我的意思了。对于一些专业的黑客来说，手里有几万个肉鸡很正常，所以群发很方便*/
SharpPcap.SetLinkConnection(可以借用肉鸡的ip地址进行处理);
String content = request.GetResponseStream();
/*如果作为一个多线程的应用程序对对方的Web构成批量发包的话（假如是DedeCms），足可以把DedeCms的数据库搞垮*/
```

H.数据安全性分析

HTTP协议的数据包都是明文传输的。也许你说“我们可以加密啊”，但是对于以上问题来讲，加密的过程就可能是明文了（比如我们知道的QQ，可以很容易判断它的加密，并有效地写一个一样的加密和解密方法出来）。当站点流量不是很大的时候没有人会在乎，但是流量上来之后，那么所谓的外挂、所谓的群发就会接踵而来（从QQ一开始的群发就可见端倪）。也许我们可以很得意地说：“我们可以采用更高级别的判断甚至HTTPS来实现，”但请注意，当你做这些处理的时候，付出的将是海量的database、I/O以及CPU的成本。对于一些群发，基本上是不可能的。笔者已经可以实现对于百度空间和QQ空间的群发了。大家愿意的话可以试试，实际上并不是很难。如果采用了更高级的验证方案，比如验证码，对用户体验又是一个很意外的影响，我们是否也非常厌烦腾讯漫天飞舞的验证码呢？

I.数据同步和集群处理问题

当数据库服务器不堪重负时，我们就需要做基于数据库的负载和集群了。这时可能会遇到最让人困扰的问题：根据数据库的设计不同，数据基于网络传输时会发生数据延迟。这是很可怕的问题，也不可避免。由此，我们就需要通过另外的手段来保证在这延迟的几秒或者更长时间内，实现有效的交互。比如数据散列、分割、内容、异步处理等问题。

J.Open API以及数据共享

Open API已经成为不可避免的趋势，从Google、FaceBook、MySpace到海内、校内，都在考虑这个问题，它可以更有效地留住用户，并激发用户更多兴趣，让更多人帮助你做最有效的开发。这个时候，一个有效的数据共享平台——数据开放平台就必不可少了。如何在开放接口的同时，保证数据的安全性和性能，又是一个我们必须要认真思考的问题。

K.大量LIKE、OR、IN以及多表查询带来的性能屏障

Or、in、多表查询会造成全表遍历，如果涉及多个or查询的活，数据查询会成为另外一个瓶颈。特别是在多对多关系漫天的SNS站点来说，更需要一个有效的处理方案。

L.大量上传文件攻击

在大多数情况下，我们会将用户上传的数据分别传入不同的FileServer中。传统的处理方案会造成安全方面的隐忧，除非我们是在一组Cluter Server中，否则我们将无法共享Session。一般我们用Cookie进行处理，而Cookie对于客户端是透明的。或者我们加密处理一下，但是在Web下伪造这些数据很方便。我们可以由此很简单地构造一个攻击工具。我们可以对上传做一些限制，判断一些权限以及判断一下上传的频率，但是当数据量很大的时候，我们需要更多考虑：如何建立一套更高效的、性能更好的判断处理方案。

解决方案

要解决问题A，我们先讲解一下电信公司ADSL的布局方案，我简单用文字描述一下流程。

ADSL用户A输入用户名、密码远程登录到账户数据库（在天津），账户数据库连接计费数据库并返回状态。如果成功，连接PPPOE服务器，并进一步连接计费数据库认证服务并连接。

这里没有什么特别的地方，和QQ通讯服务是一样的，就是采用了统一的用户验证服务器，同时对于用户验

证的信息数据库是只读的，但是我们从其中可以想到什么吗？

具体架构策略

对于之前提到的问题A，我们先以用户数据库为例来做解释和要求。

首先做用户量需求估算。我们暂时把用户量级别定为三种，百万级别（M）、千万界别（S）以及亿万级别（Q），并考虑用户登录验证以及查询常用的操作，对M和S进行扩充以及了解。

这种情况下，用户数据的负载其实并非可行或不可行的问题，而是如何最大化地保证查询和更新，以及如何在各个服务器之间进行数据同步的问题。这里不再讲解如何优化和索引，只介绍架构初期的方案。下面介绍的方案如果涉及全表查询，可以采用分区视图、分表索引处理，大家可以具体搜索相关资料。

对于M级别来说，现有的DBMS经过合理布局完全可以满足需求。问题的关键其实是应对I/O问题，处理方案相对简单，对数据库的FILE文件分磁盘存贮（不是分区，是不同的硬盘），根据负载量大小，我们可以适当控制硬盘的数量和文件分区的数量。

越来越多的Web2.0 站点面临架构上的严峻挑战

对于S级别，上个处理方案已经不能完全满足需求。这个时候我们需要对注册和入库的流程进行简单修改。解决方案是数据散列和分区视图的概念，具体概念大家去Google一下，在此我不做详细的说明。

常用的方案有三种。第一种是等容扩充法：在用户注册控制的基础上，保证每个库的用户容量不超过500万，超过之后入第二个库，以此类推。这个方案可以保证系统有效的扩充性，但不能保证数据被有效索引。第二种就是共区索引方案，其实和第一种方案有异曲同工之处。但是对第一种方案进行了合理的优化，按照用户名进行了分库存贮。比如我们可以建立26个数据库，按照用户名的索引来控制用户数据入哪个库。假如用户名是CrazyCoder，那么该用户名的数据存放在用户表C中，在数据存贮时可以很方便地根据用户名进行相应的数据查询。方案二可以有效解决数据索引问题。方案三是一个更具模型化的方案，是方案一和方案二的结合，进行用户ID的编码，不是Indentify Column。我们用一种序列化的方案将用户名以编码的形式存贮，比如用户名是CrazyCoder，我们的编码方案就是通过算法进行数字化，将CrazyCoder按照C，R，A，……存贮为数字索引，然后进行分区存贮。数字类型的数据在数据库中可以更有效地被查询、更新和共享，这就是结合方案一和方案二的方案三。

对于Q级别。数据量已经足够海量了，此时无论用哪种方案都是一个让人头大的数据，不能简单地用查询的方案来处理了，可以参考S级别。但此时我们采用的方案，是根据用户活跃度的权值结合数据量进行临时数据表的存放。如果是在非意外的数据情况下，每天登录的用户量不会上千万。这个时候我们需要做一个简单的数据代理程序。一个临时的用户验证数据库，每天执行一次批处理，将活跃度高的用户账户提取到临时数据库中。查询的时候先查询临时库，如果没有，再进行全库查询。这样根据系统的负载情况来估计阈值，不同的系统，估算方案也不尽相同。

上面对于M，S，Q三种级别进行了简单概述，下面介绍一个在其之上更高级的一个查询方案——数据缓存服务器，我们也可以把它理解为缓冲服务器，数据作为只读来使用。

因为涉及到海量，数据库常规的缓存方案已经不符合我们的要求了，我们需要一个有效的缓存方案。这个时候的处理流程，其实就是将最常用最直接的数据直接存放在缓存服务器中，而这个缓存服务器定时从主服务器获取并更新信息。这是一个简单的查询，我们还可以更深入地将缓存服务器做二次缓存，也就是一次性处理输入并存放到LIST数据中，作为全局变量放到内存中进行查询，同时用散列表或者数组进行数据组索引（可以是多级），根据查询分布到各个变量中，直接从内存中读取数据。

以笔者的经验来说的话，对于Item数据不超过10K的来说，每个列表最佳的存放范围是0到6万之间。

这里先介绍一下问题的大概。假如是一个大型博客，比如对于文章和标签，每个文章可以有多个标签，而每个标签下又会有多个文章，那么数据量将是文章数乘以标签数，这个时候如何进行处理并有效的索引，将是后面要介绍的内容。其次，如果有一部分数据涉及复杂的计算处理，比如对文章进行自动索引、分类入库、文章词组标签化、关键字过滤，以及自动修正方面的处理，有些时候我们的字库很海量，传统的处理方案会让用户等待很久，而且容易引起线程阻塞，以后的文章也将就这一点为大家介绍具体的处理方案。■

■ 责任编辑：郑柯 (zhengke@csdn.net)

服务器负载均衡架构之应用层负载均衡

——服务器集群的高扩展性和高可用性

■ 文 / Gregor Roth 译 / 李明

本文的第一部分介绍了传输层服务器负载均衡的解决方案，例如基于TCP/IP的负载均衡器等等，并分析了这些方案的优缺点。TCP/IP层的负载均衡将TCP连接分散到服务器集群中众多真实的服务器上。通常情况下，特别是对于静态的网站而言，采用这种方案就可以满足需求了。不过，动态网站常常需要更高层（相对传输层来说）的负载均衡技术。例如，如果服务器端的应用必须处理缓存或者应用会话数据，那么如何有效进行会话保持就要作为考虑重点了。在第二部分中，我将讨论如何在应用层实现服务器负载均衡，以满足众多动态网站的需求。

中间服务负载均衡器

相对于低层负载均衡方案，应用层的负载均衡需要与应用的相关背景结合。比较流行的负载均衡架构同时包括一个应用层负载均衡器和一个传输层负载均衡器，如图1所示。

图1 传输层和应用层相结合的负载均衡

应用层负载均衡器对于传输层负载均衡器来说，就像是一个普通的服务器。传入的TCP连接会转发到应用层的负载均衡器。当应用层负载均衡器收到一个应用层的请求时，就会根据应用层的数据来决定把请求发送给哪个目标服务器。

CodeFile_1.java所示的就是一个应用层负载均衡器，它通过HTTP请求参数决定使用哪个后端服务器。与传输层负载均衡器不同，应用层负载均衡器是根据具体的应用层HTTP请求来决定路由，而转发的单位就是一个HTTP请求。与第一部分里讨论的memcached方法类似，这个方案也使用了基于散列值的分区算法来决定由哪台服务器处理请求。像用户ID或会话ID这样的属性常常用于作为划分的键值。这样的话，同一个用户的请求都是由同一台服务器进行处理的。也就是说，这个用户与这台服务器关联了起来，更形象的说，是“粘”到了这台服务器上。因此，服务器就可以利用第一部分讨论到的本地HttpRequest进行缓存了。

在CodeFile_1.java中，LoadBalancerHandler读取HTTP请求的id参数并计算散列码。有些情况下，负载均衡器必须读取（部分）HTTP包体，来获取负载均衡算法所需要的数据。一个请求转发到哪里，取决于对数据取模的结果，这是由HttpClient对象来实现的。出于性能方面的考虑，HttpClient会维护一个服务器连接池来重用（或者称之为保持）连接。而应答则是通过使用HttpResponseHandle来异步处理。这种非阻塞的异步方法，可以把负载均衡器对系统的要求降到最低，例如在调用之时便无需额外的线程。关于异步非阻塞HTTP编程更详细的讲解，请看我的文章“Asynchronous HTTP and Comet architectures”。

另一种中间应用层负载均衡技术是cookie注入。负载均衡器会检查请求中是否包含特定的负载均衡cookie。如果没有，就使用类似round-robin的分布算法选定一个服务器，并将负载均衡会话cookie添加到应答中再发送。浏览器收到会话cookie后，会将该cookie存在临时内存中，并保持该cookie，直到浏览器关闭。浏览器会将cookie添加到所有后续的请求中发送给负载均衡器。使用服务器ID作为cookie值，负载均衡器就能（在当前的浏览器的会话期间）确定由哪一台服务器来处理这个请求。CodeFile_2.java就是一个基于cookie注入的负载均衡器实现。

不过cookie注入的方法只有在浏览器接受cookies的情况下才能奏效。如果用户禁用了cookie，会话保持将会丢失。

一般来说，中间应用层负载均衡器方案的缺点在于需要额外的节点或进程。传输层和应用层负载均衡进行集成的方案虽然能够解决这个问题，但是可能会花不少钱，而且访问应用层数据的灵活性也会受到限制。

基于HTTP重定向的服务器负载均衡器

有种方法可以避免额外的网络阶跃（hop），就是使用HTTP的重定向（redirect）指令。借助重定向，服务器便可以将客户端重新路由到另一个地址。服务器并不是返回请求对象，而是返回一个类似“303 See Other”的重定向响应。客户端确认新地址后，会重新发送请求。这个体系结构如图2所示。

CodeFile_3.java实现了一个基于HTTP重定向的服务器负载均衡器。CodeFile_3.java中的负载均衡器并不转发请求，而是返回一个重定向状态码，同时包含了另一个地址。根据HTTP规范，客户端会将请求重发到另一个地址上。而该客户端的后续请求将直接发送给相关服务器，不再需要额外的网络阶跃。

HTTP重定向的方法有两个弱点。首先，整个服务器架构会暴露给客户端。如果客户端是因特网上的一个匿名用户，就有可能导致安全问题。服务提供商通常都试图隐藏服务器架构，来将可能的攻击降到最少。其次，这个方法对高可用性没什么益处。与基于DNS的负载均衡（第一部分中讨论过）类似，如果对应的服务器失效，客户端不会再转发到其他的服务器上。客户端也没有简单的方法去识别出服务器是否已经死掉，而是不断进行重试。如果客户端继续使用最初的请求地址进行调用的话，网络跳数是不变的，因为每次请求都会先到达负载均衡器，然后再进行重定向。

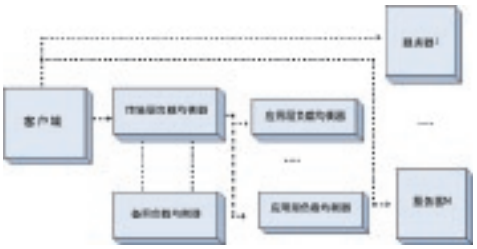


图2 基于HTTP重定向的服务器负载均衡

服务器端负载均衡拦截器

另一种避免网络阶跃的方法，是将应用层负载均衡器的逻辑转移到服务器端。如图3所示，负载均衡器会变成一个拦截器。

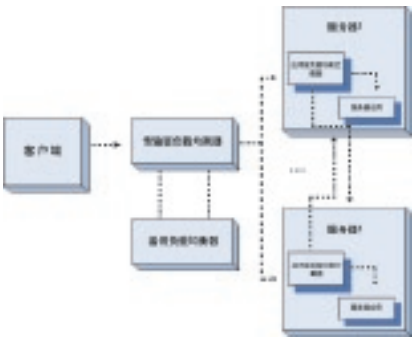


图3 服务器端负载均衡拦截器

CodeFile_4.java的代码中实现了一个服务器端应用层负载均衡拦截器。代码与CodeFile_1.java中的LoadBalancerHandle几乎完全一样。不同之处在于如果请求目标就是本地服务器，请求将本地转发，而不是使用HttpClient转发。

这种方法可以减少网络阶跃。平均来说，本地处理的请求的百分比等于100除以服务器的数量。不过此方法仅在服务器数量很少时比较有用。

客户端负载均衡拦截器

服务器端负载均衡拦截器负载均衡逻辑，通过客户端拦截器也可以实现。这样就不再需要传输层的负载均衡器。其架构如图4所示。

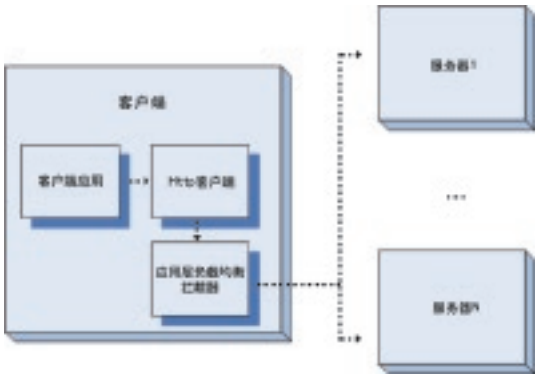


图4 客户端负载均衡拦截器

CodeFile_5.java在HttpClient中添加了一个拦截器。由于负载均衡是通过拦截器来实现的，因此，对于客户端应用来说，负载均衡是不可见的。

客户端方法有高效、高可用性及高可扩展性等优点，但是对于基于因特网的客户端来说，也存在一些很严重的缺陷。和基于HTTP重定向负载均衡器类似，整个的服务器架构都会暴露给客户端。而且，这种方法常常迫使客户端Web应用进行跨域调用。出于安全因素的考虑，Web浏览器和基于浏览器的容器，例如Flash运行时或者JavaScript运行时，都会禁止跨域调用。这就意味着在客户端必须采取某些变通方案。客户端负载均衡方法并不仅限于基于HTTP的应用。例如，JBoss支持smart stubs。stub是一个由服务器生成的对象，实现了一个远程服务的业务接口。客户端通过stub对象实现本地调用。在负载均衡的环境中，服务器生成的stub对象就是一个拦截器，它知道如何把调用路由到合适的服务器上。

应用会话数据支持

正如第一部分中所说，应用会话数据可以表示特定用户应用会话的状

态。对于传统的Web应用来说（WEB 1.0），应用会话数据是保存在服务器端的，如CodeFile_6.java所示。

在CodeFile_6.java中，应用会话数据（容器）需要通过getSession方法来获取。当传入参数为true时，如果会话不存在的话，就会创建一个新会话。根据Servlet API，一个叫做JSESSIONID的cookie将会发送到客户端。JSESSIONID的值即为唯一的会话ID。这个ID用来表示会话对象，而这个对象是保存在服务器端的。当服务器收到后续的客户请求时，就会根据客户请求的cookie头得到相应的会话对象。为了支持拒绝cookie的客户端，服务器采用URL重写（URL rewriting）来进行会话跟踪，每个响应页面的本地URL被动态重写成包含会话ID的URL。

与缓存数据不同，应用会话数据并不是一份冗余数据。如果服务器崩溃，应用会话数据就会丢失，而且绝大部分情况下是不可恢复的。因此，应用会话数据要么得全局存储，要么需要在相关的服务器间冗余保存。

如果数据是冗余的，那么通常所有相关的服务器都需要持有所有会话的应用数据。因此这种方式只能适用于小规模的服务器集群。因为服务器内存是有限的，而且更新数据时也必须将数据复制到所有相关服务器上。要想支持更大规模的服务器集群，就必须将这些服务器划分成更小的服务器组。相对于冗余所有数据的方式，全局存储方式则是使用数据库、文件系统或者内存会话服务器，在某个全局位置上存储所有的会话数据。

一般来说，应用会话数据的处理不会强制客户端关联到服务器上。如果采用冗余数据的方式，通常所有的服务器都会持有全部的应用会话数据。一旦会话数据发生变化，所有的服务器上都要更新。而在全局存储方式中，则是先获取应用数据再处理请求，而发送应答时会将会话数据的变化回写到全局存储器中。这个全局的存储器必须具有高可用性，因为它是整个系统的热点组件之一。一旦存储器不可用的话，服务器将不能处理请求。

然而，因为进行了会话保持，便可以更容易地同步相同会话的并发请求。对于多线程访问时的会话状态管理问题，请详见“Java theory and practice: Are all stateful Web applications broken?”。此外，如果客户端与服务器产生关联，就可以实现一些更高效的技术了。例如，如果使用了会话服务器的话，那么其职责就会退化为仅用于备份而已。其体系结构如图5所示。会话ID通常都用作此类体系结构中的负载均衡键值。

图5 基于备份会话服务器的应用会话数据支持

当应答返回时，应用会话数据的修改会更新到会话服务器中。和非客户端关联方式相比，服务器只有在处理失效备援时，才会读取应用会话数据。

在CodeFile_7.java中，基于xLightweb HTTP库定义了一个ISessionManager，实现了这个行为。BackupBasedSessionManager类用于在服务器端管理会话。BackupBasedSessionManager类实现了ISessionManager接口，用以拦截容器的会话管理。如果本地找不到这个会话，BackupBasedSessionManager会尝试着从会话服务器获取会话，这仅仅在服务器进行了失效备援了后才会发生。一旦会话状态发生变化，就会调用BackupBasedSessionManager的saveSession方法，在备份应用服务器上存储该会话。客户端服务器负载均衡也经常需要访问会话服务器。

Tomcat负载均衡架构

你可能会奇怪：为什么在前面的示例里，都没有用到Java Servlet API？原因很简单，与xLightweb这样的HTTP库不同，Servlet API是一个完全同步的、阻塞式的API。由于Servlet API对异步和非阻塞支持的很差，因此基于它的实现效率会很低下。无论是中间负载均衡器还是服务器端负载均衡的方案均是如此。客户端基于拦截器的负载均衡不在Servlet API的作用范围之内，因为Servlet API是一个服务器端的API。那么，你能做的也就是使用Servlet API，来实现一个基于HTTP重定向的服务器负载均衡器。Tomcat 5有这个应用，叫做balancer（Tomcat 6中不包括此应用。）

有种流行的Tomcat负载均衡方法，是将Apache HTTP Server作为一个Web服务器，通过Apache Tomcat Connector（AJP）协议将请求发送给某个Tomcat实例，如图6所示。

通过使用Apache中负责处理代理均衡的mod_proxy_balancer模块，Web服务器就可以成为一个应用层服务器负载均衡器。会话保持就采用基于cookie/path的JSESSIONID参数的方式实现。我在前面提过，JSESSIONID这个cookie参数是通过在一个servlet里获取HttpSession来隐式创建的。

通过JSESSIONID中必要的路由信息，服务器就可以修改响应，来决定目标服务器。如果客户端继续发送请求，路由信息将从请求的JSESSIONID值中提取。请求会根据此信息来转发到相应的目标服务器。

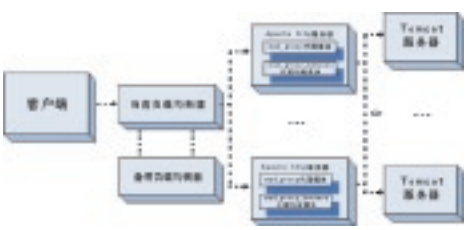


图6 流行的Apache Tomcat架构

要想使应用会话数据达到高可用性，就必须建立一个Tomcat集群。Tomcat提供了两种基本思路：将会话存放至一个共享文件系统（或数据库）中，或者使用内存数据冗余。内存数据冗余是最流行的Tomcat集群方式。开发人员也可以自行编写Apache应用层负载均衡器模块，实现多个Tomcat实例间的负载分散。或者，也可以使用其他基于硬件/软件的负载均衡解决方案，类似的方案在文章前面部分也已经讨论过。

总结

客户端服务器负载均衡是一种简单而强大的技术。它不需要中间服务器负载均衡器，客户端直接与服务器通信，但其应用范围也很有限。对因特网客户端来说必须支持跨域调用，而这又会带来复杂性和某些限制问题。

纯传输层负载均衡体系结构简单、灵活而且高效，与客户端服务器负载均衡相比，客户端也不受限制。这种体系结构通常都是与分布式缓存或会话服务器联合使用，才能够解决应用层的缓存和会话数据问题。然而，当缓存/会话服务器的数据交互增加，进而导致总开销变大时，这种架构的性能就会很快下降。通过实现基于应用层服务器负载均衡器的会话保持，可以避免服务器间的大数据量拷贝。这不仅适用于应用层服务器负载均衡，例如可以把特定的重要用户转发到指定的服务器上，而这些服务器能够提供更高品质的服务。又或者，将特定的业务功能分组转发到特定的服务器上。

虽然本文没有讨论商用和基于硬件的解决方案，但在设计服务器负载均衡架构时，的确应该考虑到它们。总得来说，选择什么样的服务器负载均衡架构，是取决于特定的业务需求和限制。

（感谢www.JavaWorld.com网站授权《程序员》杂志发表本文，原文链接：http://www.javaworld.com/javaworld/jw-10-2008/jw-10-load-balancing-2.html。文中的示例代码可从《程序员》官方博客下载，详细地址：http://blog.csdn.net/programmer_editor）■

作者简介

Gregor Roth，xLightweb HTTP程序库的作者，软件架构师，就职于United Internet组，这是欧洲主要的Internet服务提供商，客户包括GMX、1&1以及Web.de等。他兴趣甚广，包括：软件和系统结构、企业结构管理、面向对象设计、分布式计算以及开发方法等等。

译者简介

李明，InfoQ中文站Ruby社区首席编辑，开源爱好者。曾就职于百度网页搜索部，从事分布式网络爬虫的研发。目前在某通信公司任高级开发工程师，从事高性能大规模分布式系统的研发工作。爱好广泛，乐于分享。

SUN与南京大学联合打造全新人才培养模式

记者 / 周至

软件产业的发展水平，决定了一个国家的信息产业发展水平及其在国际市场上的综合竞争力。目前，我国高级软件人才的短缺已经成为制约我国软件产业快速发展的一个瓶颈，国内市场对软件人才的需求每年高达几十万人，尤其是高素质的软件工程人才极度短缺。

针对这一问题，Sun公司近日宣布，将与南京大学软件学院合作，在国内首次将Sun Java权威认证部分高级课程纳入学位教育课程体系的硕士教育，打造一套全新的人才培养模式。据悉，该教育计划将有望每年培养80到100名Java企业架构设计专业方向的工程硕士。

南京大学软件学院软件工程系主任邵栋副教授在接受采访时表示："我们这次主要培养的是在职硕士，对于入学的学员，会有一定的要求，比如他们在软件开发行业应该至少有2~3年的经验等等。此外，我们的老师会对学生在工作中遇到的实际问题进行一些指导，通过不断的实验帮助学生解决这些问题，并通过这样一个过程培养学生解决实际问题以及做研究的能力。"

媒体见面会现场

关于Sun公司在未来几年内的培训计划，Sun中国区首席教育官张瓒回答了记者的提问："今年和明年，Sun公司在培训计划方面，会有一个大的改变。以前，Sun的培训部门只针对用户（购买了Sun公司设备的客户）进行培训，而且95%的内容都是Solaris的培训。从去年开始，我们面对高校和社区提供了培训的服务，而这方面的培训95%以上的内容是有关Java的培训。所以，我们明年准备把一些Solaris的低端课程向高校推广，而把部分高端的Java课程向客户推广。"■

敏捷外包的14条原则

突如其来的金融海啸，使得全球的软件外包产业不得不重新洗牌；这对中国的软件外包企业可以算是千载难逢的机会。本文提出的基于敏捷的软件外包原则，希望能够给我们的软件人一些建议和思考。

■ 文 / Vikas Hazrati　译 / 金欣亮

虽然软件项目的外包趋势已经是一个不争的事实，但是还是有很多项目由于错误的外包而失败了。抛开诸多优势不谈，软件外包确实带来了额外的复杂性、风险以及消耗。本文将基于实际项目经验以及丰田公司的制造过程，讨论如何把外包变成一种成功的模式，我们将这种方法论称为“精益敏捷外包”。

我们的目标是利用当地的广阔人力资源，更加高效、迅速地交付软件。鉴于Scrum和丰田模式在开发新产品上的成功表现，我们决定采用它们来进行外包项目管理。丰田原则可以很好地被应用到任何软件开发项目中。我们会讨论丰田模式，以及如何使用这些原则让我们的外包过程“精益”起来。 这种保留了Scrum和丰田模式的工作方法，帮助我们每天都 在提高。

采用此方法开发的项目，是一个中心社会保障机构的批处理系统。系统的工作流程是从税务部门获取老板和雇员的数据，校验这些数据，然后生成某人某一个阶段的收入申报资料。系统每个月要处理1600万条数据记录，每秒钟大约50条数据。

原则1：制定一个长远的价值观，并且坚持下去

我们开始外包项目的时候，心中存在一个信念：我们不仅仅是为了这一个外包项目而工作，而是心中有一个长远目标：通过努力，让我们成为最好的敏捷公司，并且拥有稳定的、可重用的外包模式。这种模式可以使得整个软件行业受益。我们不会选取任何危及项目质量的捷径。 我们会积极创新，改进工作方式，提高整个行业的敏捷外包项目成功率。

原则2：创建连续的工作流程，使问题浮出水面

我们决定全面采纳敏捷实践，并且不遵从任何以计划驱动的方法论。然而，没有计划很容易迷失方向，因此频繁的交流和持续集成就变得至关重要。我们定期派遣专员进行跨越地域的旅行，让业务知识和上下文不受地域的限制，通过电话和网络进行指导，并建立和加强信任关系。 沟通不是问题，因为我们的通讯工具时刻在线，拿起麦克风就可以跟远方的团队交流了。 对于多个地点的开发，我们都采用统一的代码库并进行持续集成。这样问题很快就能浮出表面，并很快得到妥善的解决。

原则3：使用“拉”模式来避免过度生产

依照上面的原则，我们决定采用利益相关者来“拉”的模式，而不是让团队去“推”。毕竟我们的意图是编写具有商业价值的代码，而不是为了完善代码库。如果“拉力”不是很强，我们就减少迭代的大小，或者利用剩余的时间进行重构，以提升开发过程和交付的质量，确保我们高质量地满足客户的需求。

原则4：平准化

丰田提出要减少资源的浪费（muda）、人员的过劳（muri）和分工的不均（mura）。 在外包项目中，它们的表现形式为：不必要的功能、过度的需求、在客户和团队间额外的抽象层；寻找不相关的信息；测试没能找出缺陷；与客户低效率的沟通。谨记：它们是外包项目中最大的浪费。

在实际项目中，为了去除浪费，我们只开发本次迭代的用户故事，使用的故事卡片上也仅仅记录了足够开发使用的信息；我们根据用户故事进行编程，为了澄清业务细节，即使是离岸团队也可以直接跟客户取得联系；不管是开发人员测试还是客户测试，我们都秉承测试优先的做法。为了确保工作的平准化，我们总是根据团队的速度来衡量工作，并确保完成的用户故事数量没有不平衡的趋势。在软件项目中，使团队成员精疲力竭是最有害的行为！团队的速度决定了团队的工作量以及每个人的工作内容。

原则5：建立停下来解决问题的文化 （自动化）

我们的文化提倡：如果产生了影响交付质量的问题，团队就要停下手中其他的工作并解决这些问题。

如果客户处的离岸团队觉得沟通效率低下，那么我们就用专门的机器来安装skype，并保持一直在线。使用skype提供的语音和在线视频聊天功能来相互沟通。一旦团队觉得sprint评估效率太低，没有达到当初预期的目标，那么我们会中止评估，进行头脑风暴，讨论如何才能做得更好，并按照一个既定的日程来指导我们减少在代码审查上花费的时间。 如果在持续集成或

者性能测试环境设置上有问题，那么我们就要停下来修复它们，然后再进行其他用户故事的开发。 如果我们完成了很多用户故事，但是功能测试人员没有来得及进行测试，那么我们就等所有已开发的用户故事得到测试团队认可，然后再来开发新的用户故事。

所有这些不仅仅是对策，也是预防措施。

原则6：标准化过程

标准化是支撑持续改进、雇员授权、规章制度和过程的基础，同时是支持组织学习的架构。我们创建的标准流程包括：使用TDD的开发、问题跟踪和解决、构建和测试等。这并不是说这些流程都是一成不变的，它们一样是灵活且敏捷的。这些标准可以确保我们拥有稳定的平台，而且这个平台可以被团队不断完善。

原则7：使用可视化的管理来避免隐藏的问题

我们的格言是：每一件事情对团队成员都应该是可视的，当然对客户也是如此。

我们在Jira里面为客户现场和离岸团队创建了公用的产品功能清单，公用的燃尽图和问题日志。客户可以使用它来了解产品的问题，甚至查看我们每天的详细状态。使用Cruise Control可以让我们方便地了解构建的状态。每次构建结束，一个小兔子玩具会报告构建的成功与否。

墙和白板可以向客户展示足够信息。他们可以走到墙或者白板面前，10分钟的时间，就能获得足够信息。

我们还在wiki上创建了一个虚拟的团队公告板，把所有的可视化信息都保存进去。之后把有用的打印出来，贴到团队的墙上。

原则8：采用适合团队成员和过程的技术

真正的精益项目有2个关键特性：其一，它传递给开发人员最大数量的任务和职责，从而生产出有业务价值的产品。其二，它有着一个缺陷跟踪流程，保证在缺陷发生时立即处理。

一个敏捷团队最重要的资产是团队成员，我们应该让团队成员采用最适合的技术而不是听从某些技术鼓吹者的最佳实践。例如，我们把整个表现逻辑从Struts迁移到Spring MVC，是因为后者在当前的背景下更加实用，并且这个迁移的决定是由整个团队共同提出的。当团队拥有自主权的时候，也是他们能够做出最好的决定和承诺的时候。自组织的团队知道如何采用适合的技术和流程来适应每一个成员。

原则9：从团队内部发掘领导者

人比体制更重要，已经是广为人知的概念。 我们致力于从团队内部来提拔外包项目的领导者。我们会让客户现场或者离岸团队中的一部分人去参加Scrum master培训，而不是从其他地方请来Scrum master。毋庸置疑，这些人肯定是最了解团队的人。

原则10：发掘杰出的团队成员

无障碍的沟通、高效的团队协作、形式追随功能的团队、良好的收入、顶级的工具、舒适的工作环境、劳逸结合的生活、持续的进步、岗位的轮换。 所有的这些在正确的精神指导下，将会创建明星团队。

对于一个高效的跨地区团队而言，健康的交流是核心。为此，我们在项目早期会有很多探访，目的是创建良好的关系，然后用定期的探访来维持这种关系。

提出问题、谈论困难、担心无法按期限完成工作，或者对于来自上级的指令给出不同的解决方法。人们经常因为这些行为遭到责难。使团队变得更加积极主动是一场艰苦的战斗，需要花费很多的时间。因此我们鼓励多问问题。 一旦人们意识到他们有自由，同时也有责任来做决定的时候，他们会进一步做出贡献，并因此成为杰出的团队成员。

原则11：与合作伙伴和供应商建立长期合作关系

公平和互相尊重的商业关系，是与合作伙伴和供应商长期合作的关键。我们根据这个精神对客户公开WIKI和Jira，同时也对负责该项目其他模块的软件商公开。 这可以使我们清楚了解谁在做什么，并且能够同利益相关者和供应商一起来制定明确的期望。 这样做的好处是：我们建立了信任关系，保持了透明度，而且没有隐瞒任何事情。

这不仅仅提升了我们外包的可信度，从长远来看，能为软件行业提供一个扩大合作伙伴的成功实践，对软件行业来说也是有积极帮助的。通过燃尽图和Sprint迭代的记录，我们与合作伙伴互相了解和学习。

原则12：身体力行

亲临现场是了解真实情况的最佳方式。所以我们团队中没有只说空话的设计师和架构师。无论他曾经担任过什么其他的角色，每个人都要写代码，这没什么可商量的。 这有助于我们条理分明地为自己和他人分派任务，根据实际的资料请教专家，分析并理解当前的形势以及解决方案。

原则13：充分评估各种方案，达成共识之后迅速执行

我们积极地遵循一条原则：在充分考虑各种方案之前，不会选定任何方向。但是一旦我们选择了正确的道路，我们就加速并且持续地走下去。

好几次我们都试图对各种各样的issues寻求替代解决方案，比如如何使外包团队理解荷兰语文档。我们应该手动翻译还是使用一个能翻译60%内容的工具？最后我们决定让荷兰团队根据他们的文档在Jira上创建一些issue，离岸团队来研究这些issue，并且对此进行提问。这种方法非常迅速并有效。

原则14：通过深刻反思和持续改进，成为好学的组织

正是这条最重要的原则帮助我们达到了今日的高度。我们执行严格的迭代评价，深刻反思，对好的方面进行坚持，对可以提高的地方进行改善，从而使得迭代越来越好。我们渴求不断且及时地征求客户和团队的反馈，从而高效地达到软件的最终目标。除此之外，当遇到任何问题的时候，我们会一直询问“为什么”5次，直到弄清了问题的根源为止。

总结

在本文的最后，我想说：“我们仍然在学习”。“精益生产”已经被成功地应用于软件外包行业，但是仍然可以改善。丰田原则一旦应用于软件外包，将会给现有的软件外包模式带来显著的变化。我们将会通过不断改进，最终达成目标——白盒精益敏捷外包。这种模式下外包的透明度和质量都将会达到极致。

最后对外包行业提一个建议：在遵循Scrum和丰田原则的时候，不要试图稀释其中的“精华”，赶紧把它们应用到你的工作上去，然后等着奇迹发生吧。■

作者简介

Vikas Hazrati，在金融、制造、制药、健康等行业有多年的开发和项目管理经验。目前在Xebia公司担任首席咨询师。曾在多个组织中参与基于敏捷的高效开发过程和环境的搭建。

译者简介

金欣亮，28岁，现任ABB软件工程师。6年.NET开发经验，专注于企业软件架构方向以及软件开发方法学的研究。遵循以人为本的敏捷开发理念。

■ 责任编辑：郑柯 (zhengke@csdn.net)

专家点评

作为一名有多年经验，并较早在工作中实践敏捷的软件外包从业人员，笔者对该文作者提出的大部分原则可谓心有戚戚焉。下面谈一些笔者自己的体会，也算是对文中一些原则的解读吧。

关于原则1，建立长远的价值观。核心的价值观是培育健康企业文化的土壤。以笔者服务的公司为例，创始人在建立伊始，就明确了公司的核心价值是“卓越、灵活、职业和持续改进”。乍看上去似乎平淡无奇，但难能可贵的是不流于口号，而是身体力行，并取得所有员工认同。

关于原则2。作者提出的频繁交流确实非常关键。有效的交流是所有软件项目成功的必要条件，只是由于外包业的跨地域特点，交流的重要性显得尤为突出。有一点笔者或许与原作者观点不尽相同，就是文档在敏捷外包中的地位。敏捷至少不强调文档的重要性，或者干脆说代码是最好的文档。但是，设想客户团队在大洋彼岸，从未谋面（至少在项目初期），有12个小时时差。而你的团队中并不是每个人都可以非常自如地使用英语口语。在这种情况下，比通常的敏捷项目（在同一地点开发）更多一些的文档，你会不会觉得很有必要呢？

关于原则6，标准化过程。作者没有直接指出的一点是软件开发过程是否标准化，这是几乎所有外包客户考察外包公司的重要因素。所以，高效标准流程的建立是每一个外包公司必须要迈过的门槛，这也是CMM及CMMI等模型大行其道的原因。虽然敏捷思想界对CMMI有不同看法。但笔者以为，没必要纠结于学术讨论，敏捷和CMMI可以共存。当然，作者也提到流程不是一成不变的，是灵活的。毕竟，自适应性（adaptability）是敏捷核心思想之一。

原则7及原则11都谈到了对客户的透明。这是取得客户信任，建立长期合作关系的关键一点。与客户合作的过程中必然会出现问题，关键是如何解决。是回避问题，对客户百依百顺，还是态度强硬，指责客户要求不合理；是解决问题后才知会客户，还是直到问题无法避免才通知客户，或是及时与客户沟通，把问题摆在桌面上一起解决？相信大家都会有答案。

关于原则12。大一些（10人以上）的项目，一般需要全职的项目经理。要求PM也写程序，笔者认为不切实际。但笔者同意设计师和架构师要写代码，架构师至少要在代码编写上对经验少的开发人员给予指导。我们不需要只会写设计文档，画UML图的架构师。

最后，必须认识到，虽然敏捷已经开始成为主流，但并不是所有外包客户都认同或熟悉敏捷。不妨以渐进式的做法让他们逐步认识到敏捷的价值，从而取得他们的支持。比如，客户从来没有做过持续集成，与其为客户讲解该实践的好处，不如建立持续集成的环境，并给客户演示。客户看到了价值，自然会给予配合，潜移默化中，客户逐渐接受敏捷也是可以期待的。

点评专家：赵万里，南开大学计算机系研究生毕业。多年大型软件产品开发及管理经验，游走于技术和管理之间，始终保持谦卑心态，相信学无止境。

基于 Apache + Flex + PHP + MySQL 技术的流媒体网站实现方案

■ 文 / 刘江

本文主要讲述利用Adobe公司推出的RIA解决方案Flex技术实现了主题为篮球运动的专题视频网站（www.NBAvideo.com），提供了较以往类似应用更为优秀的用户体验，实现了人机交互方面的创新和实践，对视频网站的实现做了具体实践。

本系统的研发贯穿了MVC模式的设计思想，实现了数据在逻辑处理上的独立，同时，模块化的设计方法也为系统的可扩展性、可维护性提供了良好的支持。系统的具体实现涉及Flex、AMF、PHP和MySQL等多项技术，针对系统的主要功能模块介绍了如何运用合理的设计对它们进行整合。

视频网站与Flex

随着压缩技术的不断进步，视频网站的兴起使人们可以在Internet上就可以浏览高度清晰的多媒体视频，从而摆脱下载时间长、硬盘空间有限带来的问题。然而，目前流行的视频网站凭借强大的硬件设备和分布式的处理器缓存，虽然在很大程度上保证了用户的需求：延迟低、缓冲快。但用户界面日益复杂，基于HTML的页面式的交互界面，需要频繁地刷新页面难以满足用户的全方位体验。

在人机交互方面，本文采用了Flex技术来构建用户接口，Adobe公司的Flex技术作为RIA的典型代表，支持种类广泛的平台和设备，为界面设计提供了灵活多样的界面控制元素，这些控制元素可以很好的与数据模型相结合。这种用户接口，它比用HTML能实现的接口更加健壮、反应更加灵敏和更具有令人感兴趣的可视化特性。为用户提供了更加友好的交互服务和丰富的客户体验，同时减少了与服务器的响应，提高了响应速度。另外在设计中采用了MVC（Model—View—Controller）设计模式，通过模型、视图与控制器的分离良好地实现了业务逻辑和用户交互的独立，使系统的可维护性、可修复性、可扩展性、灵活性以及封装性大大提高。

系统设计方案

本系统采用了Apache+MySQL+PHP+Flex进行实现，其中系统采用PHPMyAdmin图形管理程序对MySQL进行管理维护，Flex与后台通信采用了Adobe公司推出的AMF协议。

1. 系统总体设计

图1为系统的主功能界面，本视频网站主要分为六个功能模块：登陆模块、会员信息管理模块、上传模块、视频播放模块、视频发布模块和流量统计模块。

2. 数据库设计

根据总体设计的规划，该系统需要存储用户、视频、浏览记录、视频分类、用户习惯五大类信息，建立数据库videoshare，建立相应的表：userinfo、videoinfo、record、groups、habit。

3. 主要模块的交互

下面我们结合getAllvideo()功能的数据处理流程，对所涉及的模块之间的交互做具体的介绍。该函数的功能为实现对所有视频的访问，并将视频的详细信息反馈到相应的页面。

● 模式的应用

在系统的设计实现中，都贯彻了MVC模式的应用，Flex本身的开发模式与MVC有着良好的对应关系。Flex采用了基于组件的开发模式，程序中的所有块都可以定义为组件。利用组件将界面切分开来，同时也把功能分散到每个组件中，实现了代码的封装。

另外，Flex的数据绑定功能为界面共享数据提供了方便，也可以发挥巨大作用。直接把Model中的数据作为View层的数据源，更加省心省事。

利用Remoting技术，Flex可以很方便地操作后台程序，管理数据库。把Remoting接口放在Controller中，

图1　系统的主功能界面

集成数据通信功能，一个完整的程序框架就实现了。

在这个框架中，Model依然处于重要地位，它负责储存数据，并传递数据更新的消息。总结起来，它具有三个特点：（1）存储数据；（2）支持数据绑定；（3）可以派发事件。

● 服务器端设计

服务器端程序主要负责连接数据库，执行数据库操作，并提供客户端调用。在客户端，是Controller负责和服务端的通信。对所有视频的访问必须在用户登陆成功后才能进行，用户登陆成功后，系统将存储在数据库的所有视频信息反馈到视频界面。

服务器端的函数只负责与数据库进行交互，和界面没有联系，客户端负责处理相应的数据。服务器端部署结束后，可利用AMFPHP提供的工具进行快速检测。

● 客户端设计

完成了数据库设计和服务器端程序开发后，下面进行客户端界面的开发。客户端对该功能的实现主要涉及以下5个部分：（1）主程序界面。（2）登陆界面。（3）主运行界面。（4）视频主界面。（5）视频列表。

在程序设计中，ModelLocator、userControl是处理数据的核心对象，被界面组件共享。在ModelLocator中，与getAllvideo功能相关的数据为：videos，它存放了当前的所有视频信息，类型为ArrayCollection，在ActionScript中表示为增强型数组，用来存储视频的各种信息。userControl扮演了MVC中的Controller角色，负责与服务器端进行数据通信。

4. 主要功能模块设计

● 播放模块的设计

播放模块最能体现系统的交互性，用户通过功能丰富的在线播放器可充分享受Flex这一RIA技术所带来的优越性。在Flex组件中，为网络流媒体提供丰富的用户接口和全新的用户体验，特别是对Flv这一互联网流行媒体格式的支持，使得播放器的设计更具良好的通用性，设计者通过对VideoDisplay组件的学习，可以迅速设计出功能强大的播放器，为用户展示丰富的客户体验。下图为系统的视频播放模块图。

图2 系统视频播放模块图

如图2所示，播放模块实现了播放进度条，时间显示，声音控制，按钮逻辑选择等基本功能。

● 流量统计模块的设计

流量统计模块对于网站的开发是必不可少的，通过对用户的访问流量进行统计，可以得到用户的访问分布，由此可以进一步设置更加合理的节目，提高网站的访问量，而传统的HTML这能对数据进行初步的统计，对图形显示支持不足，不能给用户以良好的直观感受。Flex提供了丰富的互动式图表和图形库，支持丰富资料的显示和交互资料的分析。图表组件可以在客户程序上动态演示，为用户提供了全新的客户体验。

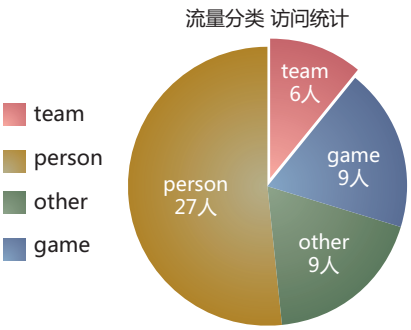


图3 流量统计模块图

● 上传模块的设计

当前的网站，已从单纯的服务商提供服务，发展为依靠广大用户本身来提供资源，视频网站的视频来源更需如此，特别是You Tube视频网站的巨大成功，验证了UGC(User Generated Content用户创造内容)、互动社区、开放平台等先进的互联网理论。对于用户上传视频有带宽、格式转换等诸多因素需要考虑，本系统只实现了视频上传的基本功能，在功能方面还有待完善和改进。

在ActionScript3.0中，和文件上传有关的类共有4个，均位于flash.net包中，分别为：FileReference、URLRequest、URLVariables和FileFilter。其中URLRequest用来指定服务端程序的路；URLVariables用来携带数据，传递给服务器端；FileFilter用来过滤文件。当打开了一个文件浏览窗口时，可以使用它来限制用户只能选择指定后缀名的文件。

总结

本文主要通过一个实际项目开发，介绍了利用MVC模式的Flex、PHP和MySQL数据库技术开发视频网站。其中，用户接口采用了Flex技术实现，为用户提供了更丰富的客户体验。

随着互联网影响的不断扩大，基于B/S架构的WEB应用得到迅猛的发展。但人们已不满足当前网页技术的效果和交互性，期待网络应用程序能够提供更好的用户体验。在此背景下诞生了RIA概念和开发模式。RIA技术是对目前B/S架构的反思和对C/S架构回归要求的背景下，产生的用户体验要求。它既有B/S架构的“零部署”好处，又有C/S架构中功能强大，表现力丰富的优点，更能满足网络应用发展的要求。而由于Flash的广泛使用和部署，Adobe公司的Flex技术在新一代网络应用中具备很强的竞争力，Flex提供了丰富的功能组件，特别是对多媒体的控制和展示提供了强大的支持，发展前景非常乐观。

小型软件公司的绩效考核

国内有着诸多小型软件公司，“如何做绩效考核”是这些公司普遍面临的问题。本文从“组织模式”、“团队建设”、“开发流程”等几个方面探讨了一些行之有效的绩效考核方法。

■ 文 / 周群

近几个月，我常和一些朋友讨论如何在小型软件公司中对软件开发人员进行绩效考核的问题，发现很多朋友都为此问题而烦恼。由于我正好在一家小型软件公司里负责绩效考核工作，有一些成功的实践经验，所以特此在这里与大家交流一下我的心得。这些心得可能仅限于小型软件公司，因为某些方法只符合小型软件公司的特点。如果你在一家大型软件企业里工作，那么这篇绩效考核的文章可能与你无关。

我这里提及的小型软件公司是指50个人以下的公司。因为从工业和信息化部提供的软件企业数量以及从业人员数量来推算，50个人的软件公司应该算是小公司了。而从《2007中国软件自主创新报告》里我们得知，在本土软件企业中，拥有50名员工以下的企业占60%。在这样的比例之下，解决如何搞好小型软件企业的绩效考核，提高企业竞争力的问题相对于大型软件企业更具有社会意义。实际上，小型软件公司相对于大型软件公司，更缺乏的是绩效考核，或者说是缺乏可量化的绩效考核方法。

组织模式

首先，工作如何量化？这取决于公司的组织结构。不同的组织结构导致了不同的工作量化方式。小型软件公司一般是小项目/小组织的特点［1998年软件工程过程改进小组（SEPG）会议对“小”做了定义。“小”被定义成“5个或更少的人进行为期3至4个月的开发”］，小型软件企业里十之八九都是项目型组织结构。比方说：A公司有15个工作人员，又有4个项目并行，最常见的就是分4拨人（每拨人包括项目经理、程序员、测试员）。表面上每个项目都有一个责任明确的“包工头”，最高领导者很省心。其实不然，项目做下来省不了多少心。一会儿报告来不及做，一会儿报告谁跳槽了，一会儿客户来投诉质量太差。而且，这样的人力资源利用率是否最高呢？其实更不然，撇开每个项目经理管理的能力差异（项目经理管理能力差异很大导致团队产出差异也很大）不提，每个项目组里的某段时期里中总有人空闲。我们企业改变这一现象的办法就是：把企业的组织模式改为职能型组织结构。这样一来，A公司有15个工作人员，4个项目并行，也仅有一个团队。其中有1个职能经理，4个项目经理，3个测试人员，7个开发人员。对于团队负责人职能经理来讲，人力资源的使用情况一目了然。

有朋友提出：一个人在多个项目里“切换”，但人不是CPU，不是多任务的，频繁切换（即使是一周换一个项目）会降低效率。可事实证明是可以“切换”的，甚至可以频繁到一个人一日多项目。就拿我们公司9月份来说，有14名员工“切换”在12个项目里（有的项目还在继续中，其中3个是较大的项目），所以，毋庸置疑“切换”的可行性。实际上，在小型软件企业中推行职能型组织管理，只需要改变一下领导者的管理理念和支撑的管理平台（后文中会有所提及）。

职能型组织的特点就是提高人力资源的使用率，对缺乏人员的小型软件公司来讲这一点非常的重要。当然，职能型组织模式还有其他管理优势。职能型组织趋向于“机械式组织”，它有利于专业化管理，有利于组织稳定，有利于集权化和正规化。如果企业做的项目规模比较小，并且是以非创新技术为重点的项目，那么“机械式组织”有利于创造高效率和低成本。

团队建设

组织模式确定之后，接下来就是团队建设。首先，我们要定义工作角色。有了角色，职权就能定义了。通常的角色有项目经理，程序开发，数据库开发，测试，技术支持等。由于采用的是职能型组织结构，所以项目经理的主要任务就是接受客户需求，进行设计和任务分解。项目经理对于项目的管理权利是非直接的，而是由职能经理负责这方面的工作，由他来决定哪些人做哪些事情，并要求何时完成任务（真正的实权在握）。

图1 部门缺陷系数变化

开发流程

我们从组织模式谈到团队建设（定义角色），而这些都是绩效考核的基础工作。接下来就是一个重点，A公司怎么依靠1个职能经理，把15个人、4个项目运作起来？这个问题首先涉及到开发流程。凡是搞软件开发的人都知道，开发流程大致分需求分析、设计，开发、测试和部署环节。如果你在一家公司里，发现几个人包揽了全部环节，那么可以不用花力气搞绩效考核。因为公司所依仗的就是这些“牛人”，就像地方政府依仗缴税大户一样，大都是免检。活生生的例子就是三鹿奶粉，由于它维持地方财政税收，自然是“免检产品”。那么如果要施行考核制度，就必须分环节和引入竞争。要把那少数几个“牛人”干的活，分成多个人来做。这样做有几个好处，从企业管理的难易程度上讲，多个人分工合作好于一个“牛人”独自做；从企业成本来讲，多个人的合计成本反而低于一个“牛人”；从项目风险来讲，一个人一个项目的风险不言而喻；从个人工作强度来讲，专业分工更节省人力；从个人的职业发展来讲，做得更专业比做得泛泛的好。于是，这样对于企业对于个人都有好处。

图2 月开发量的变化

有了角色分工就能和开发流程对应起来。比如，项目经理要完成需求和设计工作；开发人员要完成开发工作；测试和技术支持人员要完成测试和部署工作。那么，如何把大家的工作贯穿起来？我们是通过任务单。设计人员必须完成含有设计说明，预估完成工时等信息的任务单。职能经理分派任务单给相应的开发人员和测试人员。任务的分派过程由管理平台支持，职能经理基于这个平台，实现了职能型组织的管理。

通过管理平台跟踪整个开发过程，管理者就可以统计方方面面的信息了，比如个人的能力系数，缺陷系数等等，到这里便可以开始真正的“绩效”了。那么具体都包括哪些信息呢？针对设计人员角色有每月完成的任务单数、设计总工时、估计总工时、相应的开发总工时、相应的测试总工时、相应的测试总次数、相应的缺陷总数、缺陷系数和周工作量系数等。职能经理可以通过设计总工时或者周工作量系数，来了解设计人员工作是否饱和，哪个人设计的缺陷比较多，哪个人效率比较高等信息。举例，A公司的职能经理，他可以知道手下4个项目经理每月的工作情况（这里的项目经理，其实更应该称为设计人员，因为有的时候可能只有一个真正的项目经理，另外三人在做设计工作）。数据累计一定量之后，便可以知道哪个人不能完成最低设计工作量，哪个人能力比较强了。

设计完成之后，职能经理就可以派发任务单给开发人员了。设计人员对每个任务都有一个预估时间，对比开发人员的实际开发时间，可以得到一个能力系数。当然还有很多其他信息，比如开发人员每月完成的任务单数、相应的估计总工时、开发总工时、测试总工时、测试总次数、缺陷总数、缺陷系数等等。其中仅能力系数和缺陷系数两项指标就足够衡量一名开发人员了。另外，因公司而异，公司可能会有最低的开发工作量和缺陷率。通过统计，你会诧异的发现优秀的开发人员可比一般开发人员高出一倍产量，同时只有1/4的缺陷。这里要插一句，团队中的“南郭先生”会立刻显露无遗，而优秀的开发人员可以站出来大呼加薪了。

当开发完成之后，职能经理就可以派发任务项给测试人员了。因为有详细设计文档，测试人员只要按照文档进行测试，然后把缺陷编号录入到管理平台中。同样，管理平台也可以评价测试人员的工作效率，比如每月所测的任务单总数、测试总工时、发现的缺陷数量等等。这样管理人员就能够了解测试工作量是否饱和，哪个人找缺陷最拿手。

有些朋友问到关于管理平台方面的事情，比如成本。其实绩效考核的的目的是为了建设一个公平竞争的环境，找出业务水平有待提高的员工，让优秀的员工有相应的回报，让公司也能高效率的运作。对企业来讲，成本倒不是问题。因为采用了职能型组织之后，就会发现原来小公司内部还有那么多空闲时间（我们的管理平台就是在空闲中完成的）；另一方面，如果工作效率提高了，产量增加了，三个月就能收回开发成本。举例，我们公司执行之后的月开发量提高了179%，2个月的工作量相当于之前3.5个月的工作量。

再谈考核

我们一直在讲“绩效”，还没有谈到“考核”。公司光有“绩效”没有“考核”是不能提高工作效率的。如何建立奖惩制度也是一门学问，不同公司做法不同。《论语》中谈到：“名不正则言不顺；言不顺则事不成；事不成则礼乐不兴；礼乐不兴，则刑罚不中；刑罚不中，则民无所措手足”。这句话是告诉我们要如何做好绩效考核的。首先要名正言顺，有了良好的舆论环境才能立项做事，才能建立工作流程；有了工作流程，才能有奖罚标准；有了奖罚标准，员工就知道怎么做才是正确的。■

作者简介

周群，上海同济大学软件工程硕士。就职于上海麦格纳信息技术服务有限公司，任研发部经理。从事项目管理、绩效考核方面的研究和实践。

■ 责任编辑：郑柯 (zhengke@csdn.net)

活灵活现用Git——基础篇

在进行源代码版本管理时，继Subversion之后，GIT成为软件开发人员的新好。本文首先介绍Git的基本概念和原理，并穿插一些常用命令的使用。

■ 文 / 初悦欣

Git是什么

Wikipedia上称Git为强调快速度的源代码管理工具。Git最初被Linus Torvalds开发出来用于管理Linux内核源代码的开发。每一个Git的工作目录都是一个完全独立的代码库，并拥有完整的历史记录和版本追踪能力，不依赖于网络和中心服务器。

Git的出现减轻了许多开发者和开源项目对于管理分支代码的压力。由于分支得到了良好控制，开发者更愿意对自己感兴趣的项目做出贡献。其实许多开源项目包括Linux kernel, Samba, Ruby on Rails等都已经使用Git作为自己的版本控制工具。它有两点最大的好处：可以在任何地点（如地铁上）提交自己的代码，查看代码版本；可以开多个分支来实践自身的想法，而合并这些分支的开销几乎可以忽略不计。那么Git与目前最常用的SVN（Subversion）有何不同呢？

Git与SVN的不同

SVN（Subversion）是当前使用最多的版本控制工具。与它相比较，Git最大的优势在于两点：易于本地增加分支；具有分布式的特性。图1与图2可以形象展示Git与SVN的不同之处：

针对“易于本地增加分支”这一特点，图1中所示的Git本地与服务器端的结构都很灵活，所有版本都存储在同一个目录中，只需要进行分支的切换即可达到在某个分支工作的效果。而SVN则完全不同，如果你需要在本地对一些自己的代码进行试验，只能

图1 Git本地与代码服务器

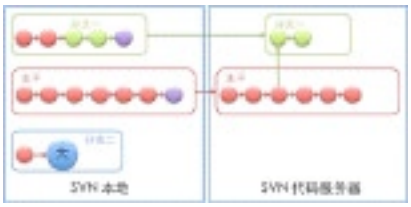


图2 SVN本地与代码服务器

在本地维护多个不同的拷贝，而每个拷贝对应一个SVN服务器地址。举一个例子来说明这个问题，以前笔者所在的小组使用SVN作为版本控制工具，当试图增强一个模块时，工作做到一半，却会由于改变原模块的行为而导致代码服务器上的许多测试失败，所以并没有成功提交代码。这时候上级说：“现在有一个很紧急的Bug需要处理，必须在两个小时内完成。”那么这时只好将本地的所有修改diff，并输出成为一个patch文件，然后回滚所有有关当前任务的代码，再开始修改Bug的任务。最后等到修改好后，将patch应用回来。如此前后需要完成多个繁琐的步骤，并且还不算中间代码发生冲突时所要进行的工作量。

可是，如果我们使用Git，只需要开一个分支或者转回到主分支上，就可以随时开始Bug修改任务。任务完成之后，只要切换到原来的分支就可以马上继续以前的任务。而且每个新任务都可以开一个分支，完成后，再将它合并到主分支上，轻松而优雅。

在图1中，Git有利于将大任务分解，进行本地多次提交，而SVN只能在本地进行大量一次性更改，增加将来合并到主干操作的风险。Git的代码日志在本地，可以随时查看，而SVN的日志在服务器上，每次查看都需要先从服务器上下载。笔者工作的小组，代码服务器在美国，当每次查看小组几年前所做的工作时，日志下载就需要十分钟，非常痛苦。后来迁移到Git之后，利用Git日志在本地的特性，笔者用Ruby编写了一个Rake脚本，可以查看某个具体任务的所有代码历史，每次只需要几秒钟，非常方便。当然，分布式并不是说用了Git就不需要一个代码中心服务器，在团队工作中还是需要一个服务器来保存所有的代码的。

Git 1+1

下面我们从Git的版本库初始化、基本操作和常用命令三部分着手，让大家掌握如何开始使用Git。

Git通常有两种方式来进行初始化：

1. git clone: 这是较为简单的一种初始化方式，当你已经有一个远程的Git版本库，则只需要在本地克隆一份就可以了。例如，'git clone git://github.com/someone/some_project.git some_project' 命令就是将'git://github.com/someone/some_project.git'这个URL地址的远程版本库完全克隆到本地some_project目录下面。

2. git init和git remote：这种方

式稍复杂。本地创建工作目录后，便可以进入该目录，使用'git init'命令对其进行初始化，Git以后就可以对该目录下的文件进行版本控制了。如果需要将它放到远程服务器上，可以先在远程服务器上创建一个目录，并把可访问的URL记录下来。然后利用'git remote add'命令来增加一个远程服务器端。例如：使用'git remote add origin git://github.com/someone/another_project.git'这条命令增加一个URL地址为'git: //github.com/someone/another_project.git'，名称为origin的远程服务器。那么，以后提交代码时只需要使用origin别名即可。

有了本地和远程的版本库，让我们来试着使用Git的基本命令吧：

1. git pull：从其他版本库（既可以是远程的也可以是本地的）将代码更新到本地，例如：使用'git pull origin master'命令将origin这个版本库的代码更新到本地的master主支。

2. git add：是将当前更改或新增的文件加入到Git索引中，也就表示记入了版本历史。这是提交之前所需要执行的一步。

3. git rm：从当前的工作空间中和索引中删除文件。

4. git commit：提交当前工作空间的修改内容，类似于SVN的commit命令，例如在'git commit -m "story #3, add user model"'这条命令中，提交的时候必须用-m来输入一条提交信息。

5. git push：将本地commit的代码更新到远程版本库中，例如：使用'git push origin'这条命令就会将本地的代码更新到名为orgin的远程版本库中。

6. git log：查看历史日志。

7. git revert：还原一个版本的修改，必须提供具体的Git版本号，例如'git revert bbaf6fb5060b4875b18ff9ff637ce118256d6f20'这条命令，而Git的版本号都是系统生成的一个哈希值。

上面的命令几乎都是每个版本控制工具所共有的，下面我们尝试一些Git独有的命令：

1. git branch：对分支的增、删、查等操作，例如：使用'git branch new_branch'这条命令会在当前的工作版本中创建一个叫做new_branch的新分支；使用'git branch -D new_branch'就会强制删除叫做new_branch的分支；使用'git branch'这条命令就会列出本地所有的分支。

2. git checkout：Git的checkout有两个作用：其一是在不同的branch之间进行切换，另一个功能是还原代码。例如：使用'git checkout app/model/user.rb'这条命令就会将user.rb文件从上一个已提交的版本中更新回来，而未提交的内容会全部回滚。

3. git rebase：用图3解释会比较清楚一些，当rebase命令执行后，实际上是将分支点从C移到了G。这样，分支也就具有了从C到G的功能。

图3 rebase的前后对比参考

4. git reset：将当前工作目录完全回滚到指定的版本号。假设如图4，我们拥有A-G五个已提交的版本，其中C的版本号是'bbaf6fb5060b4875b18ff9ff637ce118256d6f20'。但当我们执行了'git reset bbaf6fb5060b4875b18ff9ff637ce118256d6f20'这条命令后，结果就只剩下了A-C三个已提交版本。

图4 reset的前后对比参考

5. git stash：将当前未提交的工作存入Git工作栈中，等时机成熟的时候再应用。这里暂且提一下这个命令的用法，在后面的技巧篇中会对它进行重点讲解。

6. git config：利用这个命令可以新增、更改Git的各种设置。在后面的技巧篇中会利用这个命令个性化设置你的Git，为你打造独一无二的Git。

7. git tag：可以将某个具体版本打上标签，这样就不需要记忆复杂的版本号了。例如：你可以使用'git tag revert_version bbaf6fb5060b4875b18ff9ff637ce118256d6f20'这条命令来标记一个被你还原的版本。以后想查看该版本时，可以使用revert_version标签名来代替复杂的哈希值。

Git之所以能够提供方便的本地分支等特性，与它的文件存储机制有关。Git存储版本控制信息时使用它自己定义的一套文件系统存储机制，在代码根目录下有一个.git文件夹，会有如图5所示的目录结构。

其中，HEAD文件用于存放根节点的信息，其实目录结构就表示一个树型结构，Git采用这种树形结构来存储版本信息，那么HEAD就表示根。

refs目录存储了你在当前版本控制目录下的各种不同引用（引用指的是本地和远程所用到的各个树分支的信息），有heads、remotes、stash、tags四个子目录，分别存储对不同的根、远程版本库、Git栈和标签的四种引用，读者可以通过命令'git show-ref'更清晰地查看引用信息。

logs目录根据不同引用存储日志信息。因此，Git只需要代码根目录下的.git目录就可以记录完整版本控制信息，而不是像SVN那样在根目录和子目录下都有.svn目录。 ■

```
COMMIT_EDITMSG  HEAD            branches/       description     index
logs/           refs/
FETCH_HEAD      ORIG_HEAD       config          hooks/          info/
objects/
```
图5 .git文件夹的目录结构

《SERU需求过程框架系列文章之四》

需求沟通中的艺术

在需求获取过程中，讲究艺术技巧的沟通方式可以带领大家跨越甲方和乙方的鸿沟，抓住问题的本质，为项目成功奠定坚实基础。本文中的几个故事和技巧就明确体现了这一点。

■ 文 / 徐锋

沟通是一门艺术。而在需求沟通中，双方的专业背景通常是不同的，这就更需要应用一些得当的“艺术性技巧”。接下来，我们就通过几个场景实例来体会一下这其中的奥妙。

选择沟通对象关注的话题

对于“需求分析人员是用户与开发人员之间的桥梁”这一观点，想必大家都不陌生，但“需求分析人员是管理层和用户之间的桥梁”的观点却没有得到足够的重视。相信大家都会遇到过这样的现象：在项目的启动大会时，高层管理人员出席的时间通常很短。这看起来似乎并没有什么不合理的地方，毕竟他们工作很忙。但是，这个细节实际上暗藏着很有价值的信息，那就是很多项目在沟通过程中并没有很好地回应这部分人的核心关注点。

所以，如果需求人员能够抓住这部分人的关注点，从而完成与此类人员的沟通，那么将给项目带来十分积极的效应。下面的这个小案例就从侧面印证了这个道理。


案例&场景：

手机分销商李总觉得自己的生意还没有到人为管理不了的程度，几年来他拒绝的信息化系统厂商不计其数。所以当信息系统开发商的销售代表老马找到他时，他又要下送客令，但此时老马说道：“您近几年的利润率在下降！”。

“哦？您请坐。”，李总被这句话吸引住了。

“通过我的了解和分析，发现了一个对贵公司利润率产生负面影响的重要因素”，老马打开一张图：

“您为了鼓励门店多销售利润率高的产品，会在销售这些产品时给门店以更高的利润。但有些二级手机厂商为了提高自己的销售量，会打破这种规则，他们会直接给门店返利润，门店也愿意主推他们的手机，因为其利润要高于其他厂商的手机，而这是您不想看到的”。

“说得没错”，李总表示认同：“对这个问题，你有什么建议？”

这时老马不慌不忙地为其展示了一张产品销售分析报表，然后说：“当然，要解决这个问题的前提是找到影响利润的这个产品！”，边说着边开始在报表上指点起来。

“太好了！”，李总也顾不上掩饰自己内心的激动，“那么，如何整理出这张报表呢？”。

老马说道：“使用我们的软件系统就能够生成这样的报表，为您的决策提供最真实的依据。”

故事的结局就是这位一度顽固不化的李总花了一笔钱购买了这套系统，整个系统实施过程非常顺利，因为这解决了他一直以来未能解决的难题。

从上面的例子中我们可以得出一个道理，高层用户关心的是“问题/机会”，通常不愿意细化了解系统的内部。中层用户关心的则是“流程”、“可管理性”、“最终效果”，而操作层用户则关心“业务活动”、“操作便捷度”、“运行速度”等。

引起对方兴趣

在上一篇文章中，我们已经提到过“转换关注点”的技巧。在此，我们再介绍一个有趣的沟通场景。


案例&场景：

笔者在香港迪士尼游玩时观察到一个管理细节：在太空船游戏区中，管理人员会给每个人发一个牌子，上面画着船的号码、入口位置、你将坐的位置、以及进入入口之后的行走方向（如图1所示）。

回来之后与一位做游乐场的朋友谈起这个事情。他却回应：“吃饱了没事干呀，有必要管到这么细吗？”

我接下来告诉他：“我当时用手表计算了一下时间，每换一船人花掉的时间控制在2分50秒以内，我们可以现在到你那边再看一下”。

结果，我们发现在他这里，每换一船人花费的时间在5分钟左右。

接下来，我立刻趁热打铁地说到：“你看这样的话，一天要少赚多少钱呢？”这时这位朋友说：“我马上去做一些这样的牌子……”。

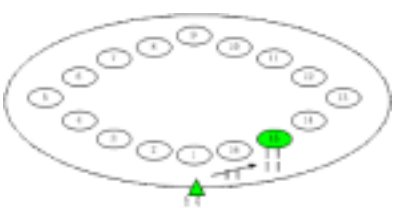


图1 路线指引牌示例

为什么这位朋友前后的态度判若两人呢？因为前面的沟通听起来是“管理的成本”，后面的沟通则涉及到了“经营的效益”。信息系统很多时候都给人一种“管理成本”的印象，但如果能够多从“经营效益”的角度来进行沟通，一定会达到更好的效果，这也是高层管理层真正感兴趣的地方。

抓住问题本质

问题经常会被表象所掩盖，如果不能在问题定义时揭开这个表象，那么经常会给解决方案带来问题。而对此类技巧阐释得最完美的当属温伯格先生的大作《你的灯亮着吗？—发现问题的真正所在》。书中收录了十余个很有意思的故事，而书名正是其中最有代表性的一个。下面我们就简要地回顾一下这个故事，看看它对软件开发工作能有什么样的启示：

隐喻：

日内瓦湖上的山脉中建成了一条很长的汽车隧道，为了防止停电时发生灾难，必须提醒司机在进入隧道之前打开车灯。针对这个问题，大家提出了解决方案：立出写有“警告！前有隧道请打开车头灯”的标示牌。但不久之后，路政部门接到投诉：隧道出口风景很美，返回时却发现汽车没电，原因是忘记了关掉车头灯。

这个问题看起来并不难解决，有人马上提出了一个解决方案：在出口处立一个标牌，写上“关掉车灯”。

问题解决了吗？当然没有！如果夜行车经过隧道时，看到“关掉车灯”的标题应该如何处理？难道真的关掉车灯，那是多么危险的行为呀！在软件开发过程中，经常也会出现类似的“惯性思维”，看上去解决了问题，实际上却带来另一个更麻烦的问题。

案例&场景：

在电子政务项目中，用户要求对每个政务申请的各种处理进行时间记录。但由于他们选择C/S结构，因此取时间时就遇到问题——每台机器上的时间都不尽相同。

“不就是时间不统一吗，让所有客户端登录时先从时间服务器上取一个时间就搞定了！”。

但这个方案在实际的运行时却带来了不小的麻烦，由于时间服务器写得不够稳定，系统经常会自动退出，严重影响了客户的正常使用。

其实解决这个问题的方法有很多，例如在数据库存储时处理，即取数据库服务器的当前时间。而且当采用这种方法时，也不要在时间服务器同步失败时阻止用户登录系统。

隐喻：

既然“关掉车灯”的解决方案是不可行的，那么就换个思路吧：有人提出建设一个充电站，以进行事后补救。但是这也有问题，不仅维护开支很大，而且本身也会出故障。

“用一个成本很大的解决方案去弥补自己的错误”是很常见的问题。原本不是什么大问题，却花费了巨大的成本来进行弥补。这种现象在软件开发过程中也不少见：

案例&场景：

在小陈负责的CRM项目中，用户认为：客户数据库的数据比较乱，有重复记录存在等现象。小陈承诺为他们开发数据清理工具，自动识别出混乱的数据，并且提供合并、汇总、删除功能”。随着这个功能的开发，项目的范围也不断扩展，针对这个功能的需求也层出不穷。这就是软件开发过程中的“充电站”，成本付出了，但真的对项目有好处吗？

这样做合适吗？似乎很多人会举手赞成，但是它将付出巨大的成本。如果我们深究一下，实际上根源问题是用户在输入数据时，系统的自动校验不足。因此，更科学的方法是加强在输入时的数据校验，而不是开发一个大“充电站”来进行事后的补救。

隐喻：

在万般无奈下，有人提出了一个蹩脚的主意：在出口立一个大牌，上面写上“如果是白天，并且车灯开着，请熄灭车灯；如果天色已晚，并且车灯没开，请打开车灯；如果是白天，并且车灯没开，就别打开它；如果天色已晚，并且车灯开着，请别关掉它”。

可谁愿意在汽车行驶过程中读完这么长且烦琐的一段话呢？

有些时候，软件开发人员也会采用类似的提示语，例如安装过程中的向导就是这样的例子：明知道大家都闭着眼睛点击“下一步”，那为什么还要不断地重复这样的设计？这难道不就是这个蹩脚的标牌吗？

那么怎样才能解决这个问题呢？关键在于对问题的定义，首先确定这里到底存在什么样的问题。如果从这个角度来看，不难发现：

表现出来的问题是车没电了，而为什么没电了呢？因为司机忘了关大灯。为什么会忘了关大灯呢？往往是没有人提醒他而导致他造成了疏忽。通过这样的分析，我们就知道需要的解决方案是“提醒机制”，这时就不难得出有效的解决方案：

隐喻：

最后终于有一个人提出了一个有效的解决方案，那就是在出口处立一个标牌，上面写上“你的灯亮着吗？”。

其实这个解决方法并不难想到，但为什么会绕了一圈呢？关键就是没有正确地认识到问题核心。这个案例诠释了“对问题进行了正确的定义，意味着成功解决了一半”的内涵。■

作者简介

徐锋，中国系统分析员顾问团软件工程首席顾问，中国软件技术大会杰出贡献专家，资深咨询顾问。主要研究领域为需求工程、系统分析与设计、软件估算，致力于推动软件工程方法论的落地应用。本文改编自笔者新书《软件需求最佳实践：SERU过程框架原理与应用》。

■ 责任编辑：郑柯 (zhengke@csdn.net)

六西格玛和软件开发（二）

继上期文章之后，本文首先从科学严谨性的角度对六西格玛进行了分析，接下来分析指出了严谨的科学方法并不一定适用于所有的决策。

■ 文 / Dave Nicolette　译 / 舒克

软件开发是业务过程吗？

六西格玛的目的，是要在业务过程中控制质量。当公司要应用六西格玛或是其他基于统计的质量控制方法时，人们总是试图将新方法运用到组织中的每一项活动之上。然而这些方法的目的，是要保证业务流程的质量。如果被用在不是业务流程的活动上，就会带来许多不必要的管理成本，同时不会产生任何业务价值。

随着六西格玛逐步演变成公司制度的一部分，我们必须要想清楚它是否能够运用到我们的软件开发过程之中。想回答这个问题，应该先弄清楚软件开发本身是否是一种“业务过程”。如果它是业务过程，那就可以像管理其他业务过程那样管理软件开发了。如果不是，那我们就必须要避免在其上强加毫无裨益的管理成本。

当今，许许多多的业务运营都需要软件应用支持。实际上，你要想与任何公司进行业务交往，都会用到计算机软件。客户服务代表要通过业务应用来回答你的询价，电话推销人员也需要业务应用的支持，银行出纳员需要业务应用处理你的交易。当你使用信用卡在油泵上购买汽油时，一个业务应用会进行信用卡的认证。很多包括自动化工作流流程处理的后端运营过程，需要业务应用和人工干预同时参与。在上述以及其他众多业务过程中，软件都起到了不可或缺的作用。

这是不是意味着业务应用软件本身就是一种“业务过程”呢？显而易见，答案是否定的。业务应用软件只是在业务过程中的工具，就像电话一样。电话不是“业务过程”，业务应用软件也不是。

那么开发业务应用软件的过程呢？它是不是“业务过程”？实施基于统计的流程控制方法的公司，倾向于将公司一切活动纳入方法控制之下。这对于软件开发的活动适合么？

接下来我会详细描述这个问题。简而言之，我认为业务应用软件解决方案的功能很重要，因为功能支撑了一家公司的业务运营。然而，软件构建时所采取的方法论，与业务运营和所支撑产品的质量毫无关系。从最终用户的角度来看，软件开发活动本身并不是业务过程。

六西格玛具备科学性么？它是不是真的那么重要？

Valeocon Management Consulting是我们公司在实施六西格玛时的帮手，根据他们提供的指导材料，基本的因果关系公式是：

$$Y=f(X1, X2, X3, X4,\cdots Xn)$$

其中，Y是结果（或是期望的产出），X是原因。公式中的f可能是非常复杂的函数，也许会涉及到分析一个混乱系统中出现的种种行为。但是不妨考虑下这样分析业务过程所要耗费的时间和精力，有两个明显的缺陷：首先，很容易只见树木不见森林；其次，当你完成分析的时候，也许稍纵即逝的业务机会早已杳无影踪了。

我们生存的社会，对于要解决的问题，总是更重视科学的方式。可没几个人能理解形式逻辑的科学方法，能够将其运用在自己日常生活中的人就更少了。当然，业务过程并不基于全面的分析，抑或理智、冷静的责问。通常，它们来自于小的尝试和错误，随着简单习惯的养成而逐步建立。可我们还希望把自己看成具备科学性的人。只要有人提供一种貌似很科学的解决方案，我们就倾向于接受它。辛苦的数据收集，缜密的数学公式，神秘的统计方法，所有这些，让六西格玛具备了科学的外形和味道。但是它是否真地具备科学性？我们又是不是真地关心呢？

问题4：六西格玛在科学上严谨吗？

Gitlow和Levine向我们保证：“相比以前的质量管理过程所管控的项目，六西格玛管控的项目具备更好的结构性，而且形式也更完备。“（我忍不住要想到：这些高度结构化和形式化的项目管理方法，导致了目前71%的IT项目失败率，我们的行业还对此津津

乐道。（这不禁让我想起了爱因斯坦对“疯狂”的定义：重复同样的行动，却期待得到不同的结果。不过这与此好像没太大关系？）

六西格玛理论基于统计数据的分析技巧。从赌博业到制造业，统计分析在很多行业中都证明了它的价值。在维基百科中的六西格玛条目中，提供了对该方法的科学统计基础的如下解释：

根据标准正态分布图，只有十亿分之二的曲线会落在6个标准差之外，而六西格玛鼓吹者宣称的是百万分之三点四。令人疑惑的是，这个值对应的精度为4.5个标准差，相对于制造业或服务业宣称的偏离了1.5个标准差。Mikey Harry在1980年前后引入该概念时，它的重要性基于观察和个人经验，而不是历史数据的积累。它被用来证明模型的不精确性，因为制造过程中的缺陷通常是无法对应到正态分布中的。实际上，流程总是要发生时间上的偏移，导致大部分问题都会落在正态分布曲线的一侧。因此，如果不考虑偏离，每一百万次操作的缺陷次数（defects per million operations，简称DMPO）会高于3.4次。然而，如果应用六西格玛方法论，如果流程偏移1.5个标准差，质量水平仍然可以停留在3.4个DPMO的层面上。

不过，对于存在1.5个西格玛的偏移这个假定，也有质疑的声音。Donald J.Wheeler是一位值得尊敬的质量控制专家，他认为上述假定很可笑，认为这在实践中被误用了，而且总是不准确的。通常实施六西格玛时，会直接加上1.5个西格玛到实际的西格玛计算中，这就让4.5西格玛的流程（3.4DPMO）转变为了6个西格玛的流程。这其中说明对于偏移的理解不太正确。如果使用短期的数据（也就是没有表明流程潜在偏移的数据），1.5个西格玛应该从最终的西格玛计算中移除，因为要考虑到潜在的偏移。因此，使用短期数据得到3.4DMPO，要代表长期的失败率的话，流程应该是3个西格玛，而不是6个西格玛。反过来说，如果使用长期数据计算西格玛，流程的偏移就应该已经计算在内了，也就不必再去掉多余的西格玛。

另外一种反对意见认为：选择1.5个西格玛的偏移太过武断，而且可能不准确。有人认为：之所以选择这个偏移，是出于市场推广的原因，这样就可以将整个过程命名为“六西格玛”而不是“4.5西格玛”，同时不必设定每十亿只发生两次缺陷这样不切实际的目标。不过，摩托罗拉在1985年使用的原始培训材料中指出，当样本空间大小为4时，标准差达到1.5个，才能检测到偏移；也就是说这个数字的选择并非毫无根据。

在实际操作过程中，六个标准差原则的应用没有完全遵从数学的严格定义。实际上，大家将六西格玛视为尽量减少缺陷的方法论或是思维方式。在非生产环境中就是这样用的，人们会将其类比为生产制造过程，而不是用作统计上的正态分布使用。与之类似，在生产制造过程中，对1.5个西格玛偏移的频繁误用，也反映出在工业应用中的类似态度。

所以，虽然人们将六西格玛视为质量改进的文化观念，但是在数学的角度来讲却并不严格。如果这么来看，至少在敏捷软件开发和六西格玛管理之间就没有本质上的冲突了。但是常言道：细节决定一切。

答案4：六西格玛更像是一种心态和方法，而不仅仅是严谨的科学理念。

在英语中，“six”和“sigma”这两个单词的头韵相同，而且阿拉伯数字“6”和希腊字母“σ”的样子也很像。也许“科学”就是由此而来的？不过，如果方法能够产生我们需要的结果，这又有什么关系呢？让我们认真思考一下。

问题5：我们需要严谨的科学方法来降低IT项目的失败率吗？

不妨先重新回顾下我们要达到的目标。我们真正想要达到的，是降低IT项目惊人的失败率，并保证在规定的时间和成本范围内，交付给客户符合他们需求的高质量产品。

使用六西格玛方法，问题在于耗费的时间过长。等到找到答案的时候，再想利用它就变得太晚了。在竞争激烈的商业环境中，这是无法令人接受的。

看看下面这个小案例。几年前，在我的公司里，一个借贷部门希望给销售代表提供无线客户端，让他们可以用其输入和查询借贷应用。他们认为这可以帮助大家更快地审核借贷，同时增加收入。困难之处在于，销售旺季即将在几个月之后到来；业务部门主管认为，如果错过这个时机，我

六西格玛的DMAIC过程

们就会丢失市场份额，并丧失这个重大的机会。我们选择使用敏捷的方法来实施项目，而不是用传统方式；因为传统的开发小组估计要用十个月才能完成工作（而我们仅用4个人的团队、6周的时间就搞定了）。此外，传统的开发小组希望先构建特定的底层技术架构，这样可以让解决方案更清晰，可真这么做会大大增加成本。

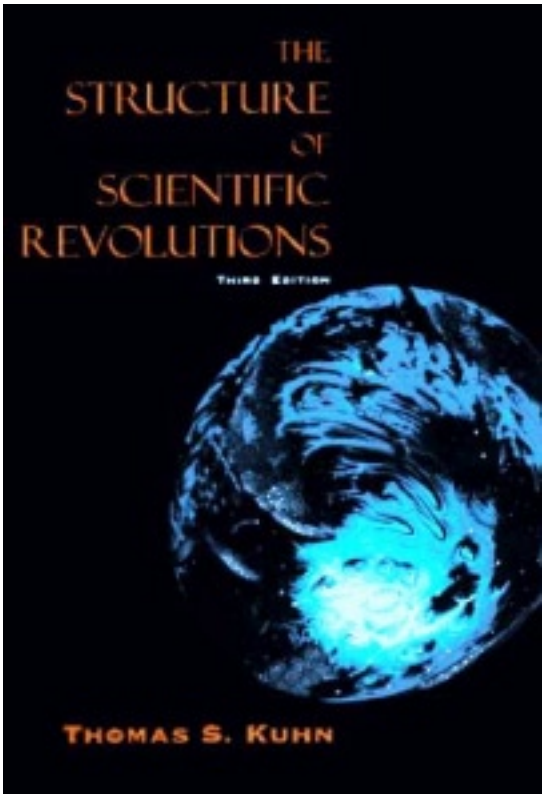
那个时侯，业务部门主管认为该解决方案能够产生营收，可没有十足把握，而且她也不想一开始就投入过多资金。根本没有时间去走需要提交大量文档的、严格的开发流程，如果真按流程走，等到所谓完美设计和实现的方案出来，早就错过市场时机了。后来发现，这个解决方案没有提高多少收入。因此，我们通过控制最少的开发成本花费，做了正确的事。

现实世界中有很多类似的问题，特别是在与环境保护和可持续发展相关的公共政策方面。如果工厂要向受保护的湿地中倾倒污染物质，而且经过过滤的水最终会作为饮食用水，要想做出公共政策决策，我们可就没有多少时间先做全面、彻底的相关因素分析了。类似情形催生了柔性科学（post-normal science）[1]。维基百科中的条目是这样的：

> “事实并不确定，价值充满争议，风险居高不下，决策迫在眉睫。”这就是典型状况。类似情形下，我们要改变硬性的、客观科学事实与柔性的主观价值观之间的关系。现在要以价值观驱动的方式做出公共决策，而不是等待收集众多的科学数据。

软件开发项目对于业务过程的支持也常常属于类似情形。这也是敏捷运动得以大行其道的一个主要因素。事实并不确定，价值充满争议，风险居高不下，决策迫在眉睫；没有时间去追随耗时、严苛、“重量级”的流程了。

上述也是六西格玛和敏捷软件开发实践之间的一个冲突点。Thomas Kuhn[2]在他极具影响力的书籍《科学革命的结构》中指出：六西格玛基于“实证科学”的思维方式。这也是“相信科学”文化的体现。



托马斯·库恩的著作《科学革命的结构》

答案5：我们在处理某些问题时，需要严格按科学办事，另外一些问题则需要“柔性”的处理方式。两种方式孰优孰劣，不能仅靠定义判定。

我们该如何调和六西格玛和敏捷软件开发？

要实施类似六西格玛的方法，组织中的敏捷软件开发人群和过程改进人群总是会存在文化上的冲突。不过，客观来看，两组人之间没有根本上的冲突。如何在花费最低管理成本的基础上满足客户定义的需求，这是敏捷软件开发的重点。六西格玛的关注点在于：如何通过应用严格的流程控制来满足客户定义的需求。差别存在于实施层面，而目标是相同的。因此，也许没有什么要“调和”的，问题可能在于想清楚何时应用哪种方案，还有要在哪些层面上将二者融入组织的标准实践。

看看敏捷软件项目的运行方式，再对比六西格玛改进现有业务流程的过程。可以发现：六西格玛的DMAIC过程与敏捷项目过程非常类似。DMAIC中的Define意味着基于客户的声音来量化期望结果，这是二者都具备的关键概念。DMAIC中的Measure是说要记录形成某些效果的具体原因，Improve就是提升的意思。而DMAIC中的Control可以与敏捷中持续学习的概念类比，在组织行为学中被称为“双回路组织学习（double-loop organizational learning）”[3]。至少从概念层面上来说，这些理念并不互相抵触或是陌生。

六西格玛和敏捷软件开发都致力于降低失误率、提升客户满意度。对于要达成的相同目标，二者体现了不同的思维方式。为了取得IT项目的成功而实施的重量级控制流程，却反而导致失败，而六西格玛要强推更为重量级的控制流程，以彻底理解为什么失败。敏捷采取了截然不同的思想：减少正式的控制，让有能力的职业人士自由发挥批判性思维（critical thinking）和创新能力，以解决独特的问题。两种方式各有其用武之地。

诸如生产制造这样的领域，彻底理解各个因素和过程之间的因果关系会带来很好的效果。在服务型组织中，也能收到成效，不过要看是否有必要减小某些因素的变化范围（要牢记：六西格玛的主要目的就是尽量减少变化）。在六西格玛培训中，常会用到比萨饼外卖服务的例子，该服务按照对客户的承诺控制送外卖的时间，同时

要保证所有的比萨饼都一致，包括外壳的高度、饼的平整性，以及外壳的颜色。我们可以通过分析所有影响到这些质量特性的因素，并找出可以改进的地方。

本文之前还提到一个餐馆的例子，虽然这两个例子都与饮食服务相关，但那是在一个非常不同的行业上下文中存在的非常不同的问题。要想选择合适的方法，就必须要理解不同的业务运营状况。如果某种方式在一种饮食服务情形下取得不错的效果，人们很容易假定它在另一种饮食服务中也能成功。要是某种方法论在一种信息技术环境下可以起作用，人们就很容易假定它在另一种IT环境中同样有益。然而事实并非如此。

软件的实施和软件开发是不同的问题。六西格玛对于软件实施项目特别有效。

统计过程控制方法已经发展了很长时间了，可以回溯到50年前戴明博士（W. Edwards Deming）所做的工作。在许多种统计过程控制方法论中，最知名的、也是用得最广泛的包括“全面质量管理（Total Quality Management）”、“改善（Kaizen）”和六西格玛等等。如果能够恰当运用在适合解决的问题上，这些方法论可以产生极大的收益。

有一个网站，专门研究如何将六西格玛应用在信息技术上。其中有一篇Gary Gack的文章（Gary Gack是Six Sigma Advantage有限公司的联合创始人之一），总结了在软件实施项目中运用六西格玛的方式。文中指出：如果在软件实施项目中应用，在六西格玛和敏捷的思想背后，有许多基本理念都是相通的。

文中有一点很有趣，而且不能忽略：作者对软件开发和软件实施项目进行了明确区分。作者声称：对于软件实施项目来说，六西格玛是非常有效的方式，而对软件开发项目则不然。这位作者有40年IT从业经验，是一名六西格玛黑带，ASQ [4] 认证的软件质量工程师，并且从沃顿商学院获得了MBA学位。

译注：

[1]：柔性科学是指，当实证科学解释再也难以招架渐趋复杂的社会结构时，开放式的公民参与以及具教育内涵的共识，将是获致最佳判断的最佳另项选择。柔性科学同时也赋予互补性的多元文化诠释观点。实证科学通常运用简化原则来提供其对于研究发现的解释，并同时强调结果的普遍适用性。

[2]：托马斯•库恩（Thomas Samuel Kuhn，1922年7月18日—1996年6月17日），美国科学史家，科学哲学家，代表作为《哥白尼革命》和《科学革命的结构》。他最有名的著作《科学革命的结构》(The Structure of Scientific Revolutions，1962年)，为当代的科学思想研究建立了一个广为人知的讨论基础，得到很多人的赞成或批评，因此他可以说是最有影响力的科学史及科学哲学家，著作也被引用到科学史之外的其他广泛领域中。纽约时报认为，因为库恩的这本著作，让范式（paradigm，中国大陆译法，台湾译作典范）这个词汇变成当代最常出现的词汇之一。

《科学革命的结构》是现代思想文库中的经典名著，作者从科学史的视角探讨常规科学和科学革命的本质，第一次提出了范式理论以及不可通约性、学术共同体、常态、危机等概念，提出了革命是世界观的转变的观点，深刻揭示了科学革命的结构，开创了科学哲学的新时期。是从事科学史与科学哲学研究的学者们不可不读的基本文献，是20世纪学术史上最有影响的著作之一，引导了科学哲学界的一场认识论的大变革，成为科学哲学史上一道重要的分水岭。其影响不仅在于科学史、科学哲学、科学社会学等相关领域，而且延伸到社会学、文化人类学、文学史、艺术史、政治史、宗教史等人文和社会科学领域。

[3]：从学习模式看。组织学习可分为单回路、双回路、三回路三种类型：①单回路学习是最基本的学习，这种学习又被称为维持学习、低阶学习、适应性学习和线性学习等。通过这类学习，组织可以发现组织策略和行为的错误，并予以纠正，使组织运作的效果能够符合组织既定的规范及各项要求。②双回路学习是指组织对既有的假设(规范、要求和目标)产生质疑，进而对之进行修正，以达到适应环境变化的目的，这类学习被称为双回路学习、创造性学习或高阶学习等。③三回路学习又称“再学习”，是指在进行组织学习时，组织成员探究过去组织学习的过程和方式，找出有碍和有助于组织学习的因素，从而提出有效的新策略来帮助提高组织学习的效率。

[4]：American Society for Quality，美国质量学会。美国质量学会是世界领先的权威质量机构，在全球拥有超过10万名个人和企业会员。自1991年起，ASQ开始负责评选颁发美国最高质量荣誉奖项——包德里奇国家质量奖。该奖每年颁发一次，用以认可并鼓舞取得卓越绩效的公司和组织。■

（未完待续，在下一期文章中，作者将进一步分析六西格玛与软件开发的关系。）

作者简介

Dave Nicolette，自1977年起从事IT行业。2002年找到敏捷后，视其为传统IT行业很多内在问题的缓解和去除之道。从那时起，在敏捷和精益的思考和实践上，他成为了一名尽心竭力的实践者和大力鼓吹的提倡者。Dave目前是美国Valtech科技公司的敏捷团队教练。

■ 责任编辑：郑柯 (zhengke@csdn.net)

DDJ006:
使用 CUDA profiler 探索全局存储器

本专栏细心的读者已经了解了之前专栏中讨论的两个反向数组示例，可能对为什么共享存储器版本比全局存储器版本速度更快仍然感到困惑。请回想一下共享存储器版本reverseArray_multiblock_fast.cu，内核将数组数据从全局存储器复制到共享存储器，然后再复制回全局存储器，而较慢的内核reverseArray_multiblock.cu只将数据从全局存储器复制到全局存储器。因为全局存储器性能比共享存储器慢100-150倍，所以慢得多的全局存储器性能占据了两个示例的绝大部分运行时。为什么共享存储器版本更快？

要回答这个问题，需要理解全局存储器的更多知识，还要使用来自CUDA开发环境的附加工具——尤其是CUDA profiler。CUDA软件的配置简单快速，因为文本和可视化版本的profiler都在支持CUDA的设备上读取硬件配置计数器。启用文本配置非常轻松，只需对启动和控制profiler的环境变量进行设置。使用可视化profiler同样很简单：启动cudaprof并开始在GUI中进行单击操作。通过配置可以了解许多有价值的信息。配置事件集合完全由支持CUDA的设备内部的硬件来处理。然而，经过配置的内核不再具有异步特征。只有在每个内核完成之后，才将结果报告给主机，这样可以最小化所有通信带来的影响。

全局存储器

理解如何有效地使用全局存储器是成为熟练的CUDA程序员的基本要求。下文简单介绍全局存储器，应该足以理解reverseArray_multiblock.cu和reverseArray_multiblock_fast.cu之间的性能差异。如果有必要，未来的专栏将会继续探讨全局存储器的有效利用。此外，可以在CUDA编程指南的第5.1.2.1节中找到关于全局存储器的详细讨论，其中包含示例说明。

只有当全局存储器访问能够合并到一个half-warp时，硬件才能以最少的事务量获取（或存储）数据，全局存储器才能交付最高的存储器带宽。CUDA Compute Capability设备（1.0和1.1）能够在单个64字节或128字节事务中获取数据。如果无法合并存储器事务，那么将会为half-warp中的每个线程发出一个独立的存储器事务，这不是期望的结果。未合并的存储器操作的性能损失取决于数据类型的大小。CUDA文档对各种数据类型大小决定的预期性能降低给出了一些简单指南：

- 32位数据类型将减慢大约10x
- 64位数据类型将减慢大约4x
- 128位数据类型将减慢大约2x

当满足下列条件时，数据块的half-warp中的所有线程执行的全局存储器访问可以被合并到G80架构上一个有效的存储器事务中：

1. 线程访问32、64或128位数据类型。

2. 事务的所有16个字所在分段的大小必须与存储器事务的大小相等（访问128位字时应为存储器事务大小的两倍）。这暗示着起始地址和校准非常重要。

3. 线程必须依次访问这些字：half-warp中的第k个线程必须访问第k个字。注意：不是warp中的所有线程都需要访问某个线程所访问的存储器才能进行合并。这称为发散warp。

最新架构（比如GT200系列设备）的合并要求比刚才讨论的架构更宽松。我们将在未来的专栏中更深入地讨论它们之间的架构差异。从本专栏的主题看，可以肯定，如果经过调优的代码能够在支持

CUDA的G80设备上进行很好的合并，那么它将能够在GT200设备上进行很好地合并。

启用和控制文本配置

控制CUDA profiler文本版本的环境变量包括：

- CUDA_PROFILE – 设置为1（或0）可以启用（或禁用）profiler
- CUDA_PROFILE_LOG – 设置为日志文件的名称（默认设置为./cuda_profile.log）
- CUDA_PROFILE_CSV – 设置为1（或0）可以启用（或禁用）使用逗号分隔的日志版本。
- CUDA_PROFILE_CONFIG – 指定最多带有4个信号的配置文件

最后一个环境变量非常重要，因为一次只能配置4个信号。通过在名为CUDA_PROFILE_CONFIG的文件中的单独行上指定名称，开发人员可以让profiler收集以下事件：

- gld_incoherent：未合并的全局存储器负载单元的数量
- gld_coherent：已合并的全局存储器负载单元的数量
- gst_incoherent：未合并的全局存储器存储单元的数量
- gst_coherent：已合并的全局存储器存储单元的数量
- local_load：局部存储器负载单元的数量
- local_store：局部存储器存储单元的数量
- branch：线程执行的分支事件的数量
- divergent_branch：warp中发散分支的数量
- instructions：指令计数
- warp_serialize：warp中基于与共享或常量存储器的地址冲突进行序列化的线程数量
- cta_launched：执行的线程块

profiler计数器注意事项

请注意，性能计数器值与独立的线程活动并无关联。这些值表示线程warp中的事件。例如，一个线程warp中的一个不连贯的存储将会递增gst_incoherent一次。因此，最终的计数器值存储的是关于所有warp中的所有不连贯存储的信息。

此外，profiler只能以GPU中的一个多处理器为目标，因此，计数器值与特定内核启动的warp总数无关。基于此原因，当在profiler中使用性能计数器选项时，用户应该始终启动足够的线程块，以确保为目标多处理器分配一致的工作比例。实际上，NVIDIA建议最好启动至少100个数据块，以获得一致的结果。

因此，用户不应该期望计数器值与通过检查内核代码所确定的数值相匹配。计数器值最适合用于确定未经优化和优化之后的代码的相对性能差异。例如，如果profiler报告软件的初始部分存在一定数量的未合并全局负载，那么很容易确定更精细的代码版本是否会利用更少数量的未合并负载。在大多数情形下，我们的目标是将未合并的全局负载数量减少为0，因此，计数器值对于跟踪此目标的实现进度非常有用。

配置结果

让我们使用profiler看一下reverseArray_multiblock.cu和reverseArray_multiblock_fast.cu。在本例中，我们将在Linux下的bash shell中对环境变量和配置文件进行如下设置：

```
export CUDA_PROFILE=1
export CUDA_PROFILE_
CONFIG=$HOME/.cuda_profile_config
```

在Linux下使用bash比较Profiler配置与环境变量

```
gld_coherent
gld_incoherent
gst_coherent
gst_incoherent
```

CUDA_PROFILE_CONFIG文件的内容

运行reverseArray_multiblock.cu可执行文件，在./cuda_profile.log中生成以下配置报告：

```
method,gputime,cputime,occupancy,
gld_incoherent,gld_coherent,gst_
incoherent,gst_coherent
method=[ memcopy ] gputime=[
438.432 ]
method=[ _
Z17reverseArrayBlockPiS_ ]
gputime=[ 267.520 ] cputime=[
297.000 ] occupancy=[ 1.000 ] gld_
incoherent=[ 0 ] gld_coherent=[
1952 ] gst_incoherent=[ 62464 ]
gst_coherent=[ 0 ]
method=[ memcopy ] gputime=[
349.344 ]
```

reverseArray_multiblock.cu配置报告

类似地，运行reverseArray_multiblock_fast.cu可执行文件生成以下输出，这些输出会覆盖.cuda_profile.log中以前的输出。

```
method,gputime,cputime,occupancy,
gld_incoherent,gld_coherent,gst_
incoherent,gst_coherent
method=[ memcopy ] gputime=[
449.600 ]
method=[ _
Z17reverseArrayBlockPiS_ ]
gputime=[ 50.464 ] cputime=[
108.000 ] occupancy=[ 1.000 ] gld_
incoherent=[ 0 ] gld_coherent=[
2032 ] gst_incoherent=[ 0 ] gst_
coherent=[ 8128 ]
method=[ memcopy ] gputime=[
509.984 ]
```

reverseArray_multiblock_fast.cu配置报告

比较这两个配置结果可以发现，reverseArray_multiblock_fast.cu不包含不连贯的存储，而reverseArray_multiblock.cu却相反，它包含很多不连贯存储。看一下reverseArray_multiblock.cu的源，并看一下是否可以修复不连贯存储的性能问题。修复之后，测量一下这两个程序彼此的相对速度。

一分钟先生

如何做时间管理?

古人云“一寸光阴一寸金，寸金难买寸光阴”，对于时间管理，古往今来的成功人士都非常重视，并且极为出色。那么我们又该如何管理好自己的每一个“一分钟”呢？

李笑来 北京艾德睿智国际教育咨询公司合伙人

http://www.xiaolai.net

为什么有的人好像一直在忙，却总是拿不出成绩，做不出成效？如果仔细观察的话，就会发现他们实际上并不努力，只是做出了努力的样子，或者显得比较努力而已。

他们的效率很差。根源在于，他们其实只做简单的事情，而回避那些有难度的工作。任何一个任务，都可以被划分为两个部分：相对简单的部分与相对困难的部分。如果这世界的所有任务全都是有简单的部分构成而全无困难之处，那就没有人会遇到挫折或者遭受失败了——可现实的世界明显并非如此。

稍微思考一下就明白，合理的时间安排应该是这样的：**简单的部分要迅速做完，而后把节约出来的时间投放在处理困难的部分上**。然而，很多人会在潜意识中回避困难，于是乎他们的时间安排是这样的：几乎全部时间都被用来处理简单的部分，至于困难的部分，干脆“掩耳盗铃”一般地视而不见，暗地里希望困难会自动消失……

如果不能控制这种逃避倾向，那么再多再巧的时间管理技巧都是无效的，因为本质上来看，任务中相当重要的一部分（通常因为重要而困难，也因为困难而重要）永远完成不了。所谓的效率需要任务完成才能够衡量，这样看来，对于逃避困难的人来讲，无论他们最终花费了多长时间，但因为任务没有完整完成，所以，根本谈不上效率（相当于分母等于零）。

雷军 金山公司董事会副主席，UCWEB董事长

http://blog.sina.com.cn/leijun

每天上班第一件事先想想今天要做什么，列出清单，先把容易的事情批处理掉。

蒋涛 CSDN、《程序员》总裁

http://blog.csdn.net/jiangtao

- 少看或者不看电视，电视提供的信息不是知识。
- 晚上早睡，早上早起可以处理很多事情。
- 多看好书，好书是智者的心得，性价比最高的学习顾问。
- 用手机阅读电子书，可以在等车乘车的时候阅读,好书还是要买纸版。

胡百敬 微软专业顾问，Ascentn大中华区架构总监

http://byronhu.spaces.live.com

有空时先想要做哪些事，排出优先顺序，用掉有空的时间。

关注IT人自我管理、职业发展，广邀经验人士排疑解惑。
花一分钟阅读，还你一个明白。
有难题吗？一分钟先生回答你！请访问http://subject.csdn.net/onemin

感谢博文视点资讯有限公司副总经理 周筠女士 对本栏目的建议和支持。

陈致平 中创软件副总经理

要想对生活和工作中的众多琐事应付自如，你必须要养成下面的好习惯：

● 每周提前规划工作和生活。可以在每个周日将下周每天要做的事情做个计划，不必100%准确，但是你可以对下周的日程有个大概了解。这可以将意外情况发生几率降到最低。根据每天的实际情况，日程可以调整，而你也不会因为错过某些事情感到遗憾。

● 要事优先。有了一周的大略日程规划，你就可以安排每天的详细工作了。早上花10分钟时间，想想今天有哪些事情要先做。很多时候，我们觉得被打搅，其实是因为没有想清楚要先做哪些重要的事情，所以总是试图在有限的时间内完成一切。现在是互联网时代，我们应该做到同时处理多项任务，但是一定要弄明白哪些事情最重要！

吕建伟（阿朱） CTO、《走出软件作坊》作者

http://blog.csdn.net/david_lv

部门管理层面

● 我一直强调方法和流程，让明确的人按照明确的处理流程做事。如果一发现新异常，没有流程和职责和明确的人负责，我就会立即把这些都补齐。

● 我不追求完美。一件事情的关键点往往就是两三个，只要不影响我的任务目标，即使出点小问题小异常，我也不爱管不爱问。我只把控事情要成功的几个关键点。

● 授人以渔。指导和训练员工，即使员工自己做不好，我也不亲自动手做，除非关键点出异常。一次听不懂，我继续在下一件事情方法上指导，直到员工自己也能做的很好。

个人管理层面

● 我清楚地知道自己的优点和缺点，总是把自己很优秀的方面练得更加优秀，根本没有为缺点投入任何精力。别人很清楚我的长处和短处，所以我不擅长的事情从来不找我。

● 我有个每日任务列表。每天要做什么事情，我都随时记录下来，然后一条条做，一条条消。

● 我以目标来检验手头的事情是否紧急和重要，如果不影响当前核心目标，这个事情就不重要。既然事情不重要，那它怎么会紧急呢。不妨大事化小、小事化了。

● 我从小爱看书，养成了快速把握重点和思路的方法，也知道如何快速鉴别这本书对自己到底有没有用。

● 我喜欢看书和思考问题，并且随时记笔记。我还经常回顾这些记录，整理成体系和方法，让彼此关联起来。所以思考问题解决问题时，总是有很多参考。而不是从头想起，也不是一遇到什么事情还得重复寻找资料。

霍泰稳 InfoQ中文站总编

http://blog.csdn.net/futurelight

根据我个人的经验，“要事第一”是我比较欣赏的。对我来说，每天打开电脑时，先问清楚哪三件事是今天比较重要的事情，想清楚了之后就马上开工，如果这件事情不需开电脑就能搞定，那么最好就别开电脑。开了电脑之后，如果没有特殊情况，IM软件和邮件统统关闭，因为做“大事”需要一个大块而且尽量无人打扰的环境。再分享一下做“小事”的经验，尤其是如何收发邮件和打电话。处理这类事情时可以采用《Getting Things Done》一书中提到的2分钟原则，即邮件看过去，如果2分钟内就能决定的，或者是简要回复，或者是直接归档，或者是移交他人，都马上处理，决不存货。这样做的原因是可以尽量保持自己脑袋的“空白”，也就是练功夫的人经常提到的“如水的状态”，他们认为人在那种情况下，威力是最无穷的。

我的数据库学习“曲线”

■ 文 / 牛新庄

编者按：牛新庄，数据库维护、优化和架构专家；曾获得国内数据库领域最高荣誉——“2006年中国首届杰出数据库工程师”；数年前曾被IBM全球软件部以年薪60万元人民币聘用，而他却婉然拒绝。这样一个躲藏在幕后的“牛人”，有着怎样的学习、发展之路？为此，本刊特邀牛新庄博士，请他讲述一个真实版的“数据库之路”。

牛新庄博士，研究方向为数据仓库和数据挖掘。是IBM官方资深培训讲师（培训DB2,AIX,MQ,WebSphere和CICS）。2002年获IBM杰出软件专家奖，2006年获“首届中国杰出数据库工程师奖”、“2006年IT168技术卓越奖”。是中信银行、山东农信、广东农信等公司资深技术顾问，中国建设银行总行特聘资深技术专家。拥有OCP，AIX，DB2，HP-UX，MQ，CICS和WebSphere等二十多项国际认证。著有《Oracle数据库开发讲座——Oracle9i Jdeveloper与J2EE实务应用》、《DB2应用开发实战指导》、《循序渐进DB2-系统管理、运行维护与应用案例》、《深入解析DB2-高级管理、内部体系结构与诊断案例》和《DB2性能调整与优化》等书。

选定发展方向

1999年，我在开始读研时就给自己确定了以后的发展方向。

当时有两个方向：网络，数据库技术。因为在2000年之时，网络大热，市场上拥有CCNP、CCIE证书的人特别牛。所以我当时也考下了CCNP证书，但后来发现网络方向涉及很多硬件层面的东西，这些都对厂商的依赖性太强，个人发挥空间不大。而我喜欢钻研，所以慢慢开始转向专攻数据库技术。

在认准数据库这个方向后，我开始深入学习数据库理论方面的知识。当时，人大王珊教授的《数据库系统原理教程》一书，我读了几十遍。在学习数据库理论的同时，我开始接触并深入学习DB2和Oracle，并从1999年开始使用DB2 V5.2。那时，市场上关于DB2方面的技术书籍几乎没有，互联网也不像现在这么发达。因为我的导师做一个课题需要用到DB2数据库，但是我只能依靠查看DB2随机文档来学习。那时，我还自己兼职，通过帮别人做些小软件赚钱，外加课题经费，以支付考OCP认证和DB2认证的费用。

到现在为止，我一直认为考认证是一个很好的学习动力。因为考试费用不菲，如果不想浪费钱只能拼命看书。我在读研的2000年就通过了OCP 8i认证，后来又陆续通过DB2 V5.2认证。这些认证极大地增强了我的自信。同时，在帮助导师用PB、Delphi等编程工具做应用开发时，我有意识地增强对SQL的学习，这对我后来的性能调优工作非常有帮助。

这里我想说的是，做好一个时期的人生规划非常重要。我们首先要有一个明确的努力方向和规划，然后有意识的往这个方向努力。这种积极主动的学习要比被动学习效率高很多。

第一次做培训

“机遇偏爱于有准备的头脑”，这句话虽是老生常谈，却是人生真谛。记得2000年底，我在网上看到一个帖子说需要一个人去安装DB2数据库，差旅报销，每天500元，我喜出望外。因为这项工作需要有DB2认证才能去，而我那时DB2高级系统管理和应用开发的认证都有，所以很快就通过了对方的审核。但是当我到客户现场时才发现，不是安装DB2而是要给客户讲课，当时我就傻眼了，因为讲课需要的知识远比安装配置数据库要难得多，更何况我之前根本没有讲过课。没办法，压力也是动力，只能前一天夜里

看教材备课到凌晨5点。短短睡了两个小时后，8点半去讲课。四天讲课下来，我总共休息了12个小时。还好自己毕竟有DB2应用开发经验和DB2认证做基础，总算勉强应付了过去。只是没想到的是，这次并不算顺利的培训，竟是我未来几年培训生涯的开始。

将培训当学习的动力

经过第一次讲课后，我看到了自己的差距，知道仅有认证是不够的。客户的很多问题，书本上没有答案，需要自己在实践经验上做努力。另外，讲课前讲师需要把一些原理、概念性的东西弄清楚，也需要对数据库进行深入学习。

后来，IBM培训部通过一些渠道知道我能讲DB2且拥有相关证书，就找我讲授DB2系列课程。所以，从2001年开始，我就经常作为IBM官方讲师讲授DB2系列的所有课程。我自认为讲课是一个很好的学习过程，因为课前要深入了解概念，对于自己的理论深入学习有很大帮助。同时，课堂上学员的实际操作问题也会强迫自己做更深入的研究。

我对培训有这样的认识：学员听你讲三个小时，要远远胜过自己看3小时的书。如果把一堂课的内容比喻成一杯水，那老师至少应该提前储备一桶水。所以，在讲课之前，我精心准备实验，深入和学员交流。我讲课从不照本宣科，而是自己准备了很多教材外比较实用的知识来扩展教材内容。同时争取上课过程中把一些概念用浅显易懂的例子来讲解。要想做到这些，首先自己必须对这个概念有深刻的理解才行，这一切都在客观上促进了自己的学习。

随着培训的增多，有部分客户开始找我做实际的调优工作。记得我第一次去为客户现场调优是2001年，去大连大通证券解决锁等待问题。客户环境用的是AIX和CICS。当时虽然问题解决了，但自己心里还是比较虚，因为对AIX和CICS不了解，万一是这两个方面有问题，自己就没办法搞定了；这让我认识到一个复杂系统的调整往往需要具备多方面的知识。这件事之后，我在网上买了一个140的IBM工作站小机，自己安装AIX并开始学习。

数据库学习Tips

根据我对数据库的理解，目前市场上虽然有Oracle，DB2，Informix，Sybase和SQL Server数据库，但Informix数据库已经被IBM收购，而Sybase数据库在技术和市场上正走向没落，占据市场主要份额的就是Oracle，DB2和SQL Server数据库。SQL Server数据库非常好，但是很遗憾的是只能在Windows平台使用。所以如果你深入研究SQL Server数据库，我只能说获取高薪的概率稍低，而且坦白的说，使用SQL Sever数据库的企业一般是中小企业居多。而国内做Oracle数据库的人太多，如果你想在Oracle领域出人头地，难度极大。但是，做DB2数据库的人反而不太多，物以稀为贵。况且，DB2数据库广泛应用在银行、电信、制造行业、零售行业、保险行业等“高薪”领域中，所以我强烈建议学习DB2数据库，做IBM技术一般获取高薪的概率相对会大一些。我们的时间精力是有限的，所以必须选择好方向然后努力为之。除了SQL Server，这几个数据库我都在使用，我个人感觉除了功能外，对于运行稳定而言，相对于Oracle不太稳定的优化器，DB2无疑是最稳定的，它的优化器无比强大。如果能在锁方面再有更先进的技术，那么DB2将是完美的。

这期间，我一边学习，一边通过了AIX的全部认证。记得非常清楚的是，为了做HA的实验，我花费了很大工夫。因为那时小型机不像今天这么普及，无法搞到7133阵列。后来我又学习了CICS、WebSphere、MQ和存储。就这样，在我培训的过程中，发现自己哪方面薄弱并且感觉这个方向有前途，我就会开始学习。不过，那时我的技术主要还是围绕IBM产品为主。由于自己对培训比较用心且颇受客户好评，找我做培训的国内培训机构开始变多。这个期间我自己的技术水平也增长很快。

2002年11月，我参加了首届“IBM DeveloperWorksLive! China 2002”大会，并获得IBM首次在国内评选的“杰出软件技术专家”奖，当时在6名获奖者中名列第2。这个奖项客观上对我在客户群的拓展方面起到很大帮助。找我解决问题的人更多了，所以2002—2003年也成了我技术提升最快的两年。

这两年内，我陆续学习了HP-UX、WebSphere和MQ并通过认证。我自己的感觉是，如果你把一门技术研究得非常深、非常透，由于触类旁通的缘故，再去学习另一门技术时就很轻松。所以，我在学完AIX再去学习HP-UX时，感觉非常轻松。同样，在学习ORACLE和DB2后再去学习Informix也同样很容易。通过这种纵向的深入和横向的比较，各种产品的所长所短也会非常清楚，自己的技术视野无意间更加全面化。而且通过对一个产品的深入，你往往能够发现这个产品的缺点和需要改进的地方。就拿DB2来说，每次版本更新的新特性，在新版本未上市前我就可以猜得差不多了。这主要有三个原因：一是我贴近真实用户，了解他们的真正需求；二是自己一直在用且不断总结思考；三是这些特性别的数据库有，而DB2没有，那在下个版本就会增加。所以相对来说，我自身对新版本的新特性学习就非常轻松了。就DB2而言，我拥有DB2 V5.2、V7.1、V8.1和DB2 V9的全部认证，而且我应该是国内第一个把DB2 V8认证全部通过的人，当然，这其中也有巧合的成分。

重要的一点是：学习过程中，要不断地把实践和理论融合，知其然更

知其所以然，这样提升就会快很多。

现场救援“赶场”记

2004—2005年是我最忙碌的两年，那时候找我讲课的培训机构和需要性能调优的客户非常多，基本上整天在天上飞。培训机构找我讲课常常需要提前一个月预约。那两年内，除了过年几天，其他时间都是在做培训和诊断、调优，足迹遍及国内主要城市。我自己基本上是国内六大银行开发中心和数据中心培训的指定讲师，并为北京银信科技、山东农信、广东农信，交行大集中IBP等项目做数据库技术顾问。

那时的我年轻、精力充沛。记得最刺激的一次是2004年9月的一天，上午9点为上海移动IT部门做AIX动态逻辑分区（DLPAR）培训，结束时是17点。之后，立刻坐出租车前往扬州，于20点到达扬州供电局并协助他们进行电力负荷控制系统项目上线，一直奋战到凌晨3点半。接着，又连夜乘出租车赶往上海，在凌晨6点到达酒店。休息两小时后，8点出发，准时出现在上海移动培训现场。那时我对报酬不太在意，想的主要是用心积累技术经验和客户资源。在我看来，能够不断通过实践让自己成长是第一要义。而且，去的客户现场越多，处理的问题就越多，也就越多地发现自己的不足，然后再拼命学习，不断积累、总结和思考，进入了一个良性循环。

至今我仍然怀念那段充实、紧张而充满激情的光辉岁月。2004年和2005年，一方面因为以独立咨询顾问的个人身份无法出具发票；另一方面，项目越做越大，尤其是很多银行的数据库架构和维护项目涉及合同金额也越来越大，需要签订正式公司合同。于是，我就分别在上海、北京注册了公司。当然这些年我并非都是一帆风顺，也犯过一些重大错误，例如：我曾经在2002年5月1日把海南美兰机场的数据库调死，导致机场航班信息管理系统瘫痪。早期也曾经因为调整某证券系统宕机而影响股民交易，这些都对客户造成了影响，但这些都是成长必须要走的路。经过这两次事件后，我自己也思考、总结了很多，在之后的调优工作中我基本上再没有犯过错误。

我的秘诀：学习、积累、规划

2006年8月我获得“2006年中国首届杰出数据库工程师”称号，算是对我多年学习数据库的一个总结。自2007年开始，我专注于做一些大客户的运维工作，并相应减少了培训次数。2008年，我被建设银行以年薪217万聘请为资深技术专家来维护Oracle和Informix数据库。就做技术而言，以一己之力能挣到年薪几百万常常令我感到自豪，也让我感受到技术的魅力，觉得自己多年来对技术的钻研得到了认可。

之所以讲述我的技术之路，主要目的是给大家一些参考，尽可能多地去了解社会的需求，有意识给自己制定人生规划。我自己认为，多年来能取得这样的成绩，勤奋、努力和坚持一直是我最看重的。因为有了这些，才不至于当机遇光顾时，你却不知所措。

现在很多年轻人，恰恰缺少的就是这样的忘我与痴迷，在我熟悉的数据库技术领域，很多年轻人越来越早地将注意力集中在薪水和职位上，这是很不明智的行为。其实，往往那些将诸如高薪与职位忘怀的人反而能更快地取得成功。“不经一番寒彻骨，安得梅花扑鼻香？”这样的道理人人都懂，可能够真正去实践的人却并不多。结合我的学习经验与感悟，我总结有16字要诀：去除浮躁，认真学习，不断积累，寻找机遇。

最后，我用这句话与大家共勉：古之成大事者，不唯有超世之才，亦唯有坚韧不拔之志也！■

■ 责任编辑：郑柯 (zhengke@csdn.net)

新 闻

Opera 桌面浏览器中国版正式上线

12月16日，Opera首次为区域市场定制的Opera桌面浏览器中国版正式上线；与此同时，Opera还将移动浏览器Opera Mini正式引入中国。Opera中国版是Opera专门为区域市场开发的浏览器，而另一款Opera Mini产品，在全球有超过2100万的活跃用户正在使用，它通过对用户访问页面的数据压缩，帮助用户快速访问Wap和Web站点。同时，为配合中国战略升级，Opera中文社区新版也正式上线。

Oracle 发布全新数据库应用开发工具

12月11日，Oracle在北京发布了全新的数据库应用工具Oracle SQL Developer和Oracle Application Express。Oracle SQL Developer是一种集成开发环境，支持多种数据库，开发人员可以用其中的可视化图形化工具创建自己的数据库应用程序，并可进行版本控制。Oracle Application Express是用于Oracle数据库的、基于浏览器的快速Web应用程序开发工具，其中提供三大功能：应用程序构建器、SQL Workshop和实用程序。

新锐国际主办“危机下成长型企业生存法则“研讨会

11月28日，由企业内容管理解决方案提供商新锐国际（GEONG）主办、赛迪顾问协办的“降低成本 积累能量—成长型企业生存法则”研讨会在京召开。此次研讨会的目的是为成长型企业如何度过此次席卷全球的经济危机出谋划策。在研讨会上，新锐国际（GEONG）发布了“成长型企业生存计划”，该计划包括三大新的服务应用：SaaS低成本软件使用模式；帮助规范管理，增加客户满意度的ISO锦囊；帮助提高赢单率的精明销售管理体验式培训。

瑞星 2009 全功能安全软件发布，开启“云安全”时代

12月16日，“瑞星全功能安全软件2009”在京正式发布。该产品基于瑞星“云安全”策略和“智能主动防御”技术开发，并将杀毒软件与防火墙无缝集成为一个产品。据悉，“瑞星2009”的亮点是“木马入侵拦截”。除“木马入侵拦截”、“网络攻击拦截”、“恶意网址拦截”等拦截模块，该产品同时也有“木马行为防御”、“出站攻击防御”等防御模块，形成了整体的安全保护体系。

C++特性约束检查（下）

本文讨论了一种允许用户定义任意代码特性集的机制，它同时能保证在编译时被调用函数满足调用者所有代码特性要求。

■ 文 / Scott Meyers 译 / 罗小平

特性约束与虚函数

在虚函数中引入代码特性将带来麻烦，这是因为C++规定在派生类中覆盖基类虚函数时，必须使用同样类型的参数。在派生类中覆盖后的虚函数，可能通过基类的指针或引用调用，这就要求前者必须提供基类虚函数要求的全部代码特性。从这个角度说，派生类中覆盖后的虚函数和基类虚函数相比，应该可以引入更多代码特性，而现实中也存在这种需求。但问题是，C++目前设定的规则不允许这样设计：

```
class Base {
public:
  typedef mpl::vector<ThreadSafe, Reviewed>
BaseFeatures;
  virtual void vf(int x, std::string& s,
MakeFeatures<BaseFeatures>::type features);
  ...
};

class Derived: public Base {
public:
  // 注意与基类特性集相比发生了变化，新增了Portable
  typedef mpl::vector<ThreadSafe, Reviewed,
Portable> DerFeatures;

// 下面的函数并没有覆盖Base::vf!
  virtual void vf(int x, std::string& s, MakeFeat
ures<DerFeatures>::type features);
  ...

};
```

我们需要一种方法，既不违反C++规则，即覆盖虚函数时使用与基类相同类型的参数，同时又能在派生类的虚函数中引入更多代码特性。如果C++允许函数参数类型的协变（基类函数特性集MakeFeatures<BaseFeatures>::type从派生类的函数特性集MakeFeatures<DerFeatures>::type继承），那么这个问题就解决了。但不幸的是C++目前并不支持参数类型协变。

重载是解决这个问题的有效办法。在派生类中声明两个同名函数，其中一个使用与基类相同的特性集类型，而另一个使用扩展后的特性集。然后，在派生类覆盖基类的函数（即使用与基类相同特性集的那个）内部调用另一个使用扩展特性集的函数。依照这种思路，上述派生类可如下实现：

```
class Derived: public Base {
public:
  typedef mpl::vector<ThreadSafe, Reviewed,
Portable> DerFeatures;  // 定义不变

  virtual void vf(int x, std::string& s,
                            // 覆盖基类虚函数
                  MakeFeatures<BaseFeatures>::type
features)
  {

     // 验证特性协变
     typedef MakeFeatures<BaseFeatures>::type
BaseFeaturesClass;
     typedef MakeFeatures<DerFeatures>::type
DerFeaturesClass;
     BOOST_MPL_ASSERT((boost::is_base_of
<DerFeaturesClass,BaseFeaturesClass>));

     return vf(x, s, MakeFeatures<DerFeatures>::
type());        // 内部调用扩展特性集函数
  }


  virtual void vf(int x, std::string& s,
      // 定义不变
                  MakeFeatures<DerFeatures>::type
features);
  ...
};
```

这种设计方法，既为通过基类接口调用虚函数的调用者提供了基类接口要求的代码特性，同时又能让它通过派生类接口照顾到派生类新引入的其他代码特性。这和C++对虚函数协变返回类型[注12]的支持比较类似。

性能

从原理上讲，特性检查并不会增加运行时消耗，因为所有检查工作都在编译时执行。但是运行时，即便不被使用，每个特性集参数也可能在调用者和被调用者之间传递——这取决于C++编译器的能力和优化设置。假设这些

特性集不被优化掉，通过虚继承[注14]，在目前实现下它们所能达到的尺寸规模见表1（每个特性集可达数千字节）。

C++ TMP（Template MetaProgramming）在增加编译时间上名声远扬，以自己的经验而言，这没有冤枉它。一个几十行代码（不包括头文件）的程序，编译常常需要30秒甚至更长时间。我在这里没有测试5个以上特性，因为编译时间太长，而且在这种情况下，Visual C++会因为出现内部编译错误而不能完成编译。

Features in Feature Set Object	gcc 4.1.1	Visual C++ 9	Comeau 4.3.9
0	64	388	7672
1	32	164	1884
2	16	68	452
3	8	28	108
4	4	12	28
5	4	4	8

表1 不多于5个特性时，几种32位编译器下sizeof(feature set)对比

不过，我的这个测试仅是为了说明原理而设计，实际上性能并不是大问题。只要更多人关注这个问题，就能设计出更加高效的实现方法。比如为了避免在已经略显累赘的TMP语法中引入更多符号，目前实现中的特性集参数是按值传递的，在未来可以改为用指针和const引用传递。

讨论

本文所讨论的技术，还可扩展到如下一些方面：

- **支持操作符**。文本已经讨论过的技术，是通过函数中特定参数实现对代码特性强制检查的。这种设计方法显然不适用于操作符函数，因为C++不允许在操作符函数中使用非操作数的参数。即便没有这个限制，怎样将特性参数集成到操作符的正常调用语法中，我们现在也没想到好的办法。例如向操作符“+”传入两个以上参数，现在使用的语法“a + b”怎么处理，或许要通过操作符重载修改现有语法？如何对操作符函数执行特性约束检查，请大家讨论。
- **函数组代码特性的指定**。Sutter博客[注20]上的一个评论，引出了这样一个问题：如何将对单个函数的特性约束检查扩展到函数集，比如同一个类或名字空间中的所有函数。理论上讲，实现这种检查仅需将特性施加到函数组上，组中的单个函数将不再需要特性集参数。
- **支持特性类之间的继承**。本文对异常安全的概念做了非此即彼的简单处理：函数要么提供异常安全，要么不提供。实际上，C++的异常安全可分为多个层次：（1）从不抛出异常（nothrow guarantee）；（2）支持事务式语义（transactional semantic；strong guarantee）；（3）仅支持不变量保留语义（invariant-preserving semantic；basic guarantee）[注2、16]。通过继承区分这些代码特性，可以建立多层异常模型：

```
struct BasicGuarantee {};
struct StrongGuarantee: BasicGuarantee {};
struct NoThrowGuarantee: StrongGuarantee {};
```

修改代码特性约束检查的方法以支持上述继承关系，应该是一件很有趣的事情。

- **取消AllCodeFeatures**。在现在的设计中，我们假设事先存在一个包含了所有特性类的容器（如AllCodeFeatures）。而更灵活的设计应该是去掉这个容器，使得所有开发者可以自定义特性类，同时要避免名字冲突，各自的代码组织起来后可无缝运行。解决名字冲突问题常用的办法是特性类自注册，但传统自注册是在运行时执行的。我们需要在编译时完成这项工作的办法。
- **允许用户控制特性集类型转换规则**。调用针对特性集类型重载的函数时，本文定下的策略是优先匹配引入多余特性的最少者。而另一个可能的策略是直接判定这种调用存在歧义（Sutter已经说明如何实现这个策略[注20]）。显然，其他策略也是可能有现实需求的。例如，用户为不同的特性指定不同的权重，要求优先匹配引入的权重和最少者。研究如何将特性集类型转换策略置于用户控制之下，应该是很有趣的——或许可以通过Policy-Based Design实现[注3]。
- **提升诊断能力**。编译器对特性约束违反行为的诊断过程还远非透明。如果能为调用函数要求的、被调用函数提供的特性集，给出简洁易懂的指示，将是一个重大进步。实现这个要求的一个办法，可能是开发一个类似于Zolman的STLFilt[注23]的后置处理器，提升STL相关诊断信息的可读性。
- **提升性能**。上文已经说过，本文所讨论技术在实际应用中可能引起编译时间和运行时内存消耗量较大上升。因此，寻找更为高效的代码约束检测机制实现是非常必要的。或者我们可以开发两个版本，一个是和这里一样的慢版本，另一个是快版本。快版本由无任何实现的API桩构成，因此可快速编译；而慢版本则用于程序的定期检查，看是否存在约束违反行为。

相关研究成果

我还没注意到在任意定义和组合代码特性方面出现其他研究成果，但在约束函数调用行为这个领域，已经有不少人关注。比如Bolten说明了如何通过使用中间类，调节对C++中友元的控制粒度[注6]；Alexandrescu已经论证在多线程系统中，可利用C++类型系统加强对数据争用条件的编译时检查[注4、5]。此外，Perl的Taint模式[注19]可对来自外部源的数据作出标记，并限制对这些数据的操作。和本文以及Bolten、Alexandrescu的论文中主要依靠编译时静态检查相比，Perl的Taint模式具有很强的动态检查能力。

总结

本文讨论了一种允许用户定义任意代码特性集，并保证在编译时被调用函数满足调用者所有代码特性要求的机制。这种机制建立在实现了C++ TMP技术的Boost MPL库之上，适用于成员函数（包括虚函数和非虚函数）、非成员函数以及函数模板，但不包括操作符。

致谢

在开展本文涉及的研究工作，包括本文写作过程中，我得到了很多朋友的支持。Boost Users mailing list成员为我学习使用MPL提供了极大帮助；Steven Watanabe无私奉献了MakeFeatures实现的精华[注21]。Herb Sutter对我初期设计方案改进和后期优化的建议，促成了最终完全实现编译时检查方案的形成[注20]。本文第一个版本出来后，Andrei Alexandrescu、Eric Niebler、Bartosz Milewski和Hendrik Schober的评论对我帮助也很大，如劝我找到避免我那时对虚函数使用的运行时检查的方法。Andrei还建议将继承作为特性集类型隐式转换的基础，并提醒我在派生类中通过重载，从而使虚函数覆盖时能支持比基类更多的代码特性。还有Steve Dewhurst，他建议我利用在主类型容器中的位置对特性类序列排序。

延续阅读

红代码、绿代码及以TMP为基础的设计方法

研究代码特性及其约束时，很容易就会涉及所谓红代码（red code）和绿代码（green code）。红代码不具有特性，因此也是无约束的，可以调用任何其他代码。绿代码具有特性，只能调用能提供同样特性的函数。

阻止绿代码在无显式指示语法（如cast）时调用红代码，我最初考虑使用的并非TMP（template metaprogramming），而是名字空间。我的想法是将红、绿代码分离到不同的名字空间；绿代码可以输入到红代码的名字空间，但不允许反向输入。这样，红代码就可以直接调用绿代码，而绿代码只能在显式引用红代码名字空间时，才能调用红代码[注15]。例如：

```
namespace green {
    void greenFunc();   // 可被其他任何代码调用，但它
//只能使用名字空间调用其他代码
}
namespace red {
    using namespace green;       // 红代码可调用任何
//绿代码
    void redFunc();              // 可调用任何代码，
//但仅能被无约束代码调用
}
void green::greenFunc()
{
    redFunc();                   // 错误！红代码在绿
//代码名字空间中不可见
    red::redFunc();              // 正确，通过名字空
//间显式调用
}

void red::redFunc()
{
    greenFunc();                 // 正确
}
```

这个办法很快陷入了泥潭。比如对于全局函数，它是不属于任何名字空间的。同时，如果绿代码函数用类型定义在绿代码空间的参数，且无显式限定语法调用红代码函数，C++的参数依赖查找（argument-dependent lookup[注11、22]）规则将导致位于红代码名字空间的这个函数被正确找到，这就绕开了我们希望通过名字空间强制执行的约束检查。此外，多个约束可能被任意组合，而名字空间只能嵌套使用，我简直无法想象通过嵌套名字空间实现任意约束的组合将是一番什么景象。

我想到的第二个办法是利用类似于Barton和Nackman在编译时检查空间单位正确性的方法[注8]。这个方法以向对象增加额外信息为基础，而我的工作对象是函数，很难找到逾越二者之间差异的办法。

然后我又想利用enable_if技术，针对特定调用控制函数是否可用[注13]。不幸的是，在我的要求与enable_if的设计目标间存在无可弥合的距离。我需要的是受约束代码调用无约束代码时，不能通过编译，但在enable_if中，当控制条件得不到满足时，是从候选的重载函数表中将此函数删除。而这并不能保证编译不被通过，因为仍可能找到其他匹配的函数。

enable_if的另一个问题是，它仅适用于函数模板，而不适用于函数。因此它不能用于虚函数，因为虚函数可能不被模板化。同时它也会造成代码膨胀，因为带有不同enable_if参数的函数模板通过多次实例化会引发同一个对象代码的多个副本。这也是我在初期设计中忽视的一个问题——在我对代码约束的初版实现[注18]中，尽管没有使用enable_if，但是仍以所有受约束函数均为模板为前提的。

放弃enable_if后，我转向特征（Trait）机制，寻找将约束信息与函数相联系的办法。特征主要用于实现类型与信息的映射，但它们也能用于建立信息和值的联系，而函数地址恰是值类型。但最终我放弃了这个思路，原因之一是函数模板难以对付（它生成多个函数，因此有多个地址），此外也因为特征将函数约束和函数的定义从物理上分离，而函数的约束是函数接口的关键组成部分。而且，函数约束直接出现在函数声明中或附近，是十分重要的。

到这个时候，我突然意识到编译时空间单位分析、enable_if以及特征技术有一个共同点：都建立在TMP基础上。我开始考虑是否可利用TMP技术，特别是MPL解决代码约束的检查问题。接着通过对STL中遍历器类别通

过空声明体类表述的研究，慢慢走到了本文所讨论技术的门口。

注释

1. Abrahams, David and Gurtovoy, Aleksey, C++ Template Metaprogramming, Addison- Wesley, 2004.
2. Abrahams, David, “Exception-Safety in Generic Components,” Generic Programming: Proceedings of a Dagstuhl Seminar, M. Jazayeri, R. Loos, and D. Musser, eds. (Springer Verlag, 1999), available at http://www.boost.org/community/exception_safety.html.
3. Alexandrescu, Andrei, Modern C++ Design, Addison-Wesley, 2001.
4. Alexandrescu, Andrei, “Multithreading and the C++ Type System,” February 8, 2002, http://www.informit.com/articles/article.aspx?p=25298
5. Alexandrescu, Andrei, “volatile – Multithreaded Programmer's Best Friend,” C/C++ Users Journal C++ Experts Forum, February 2001, available at http://www.ddj.com/cpp/184403766/.
6. Bolton, Alan R., “Friendship and the Attorney-Client Idiom,” C/C++ Users Journal, January 2006, available at http://www.ddj.com/cpp/184402053/.
7. Boost C++ Libraries Web Site, http://boost.org/
8. Barton, John J. and Nackman, Lee R., “Dimensional Analysis,” C++ Report, January 1995. This is a simplified version of the material covered in section 16.5 of the authors' Scientific and Engineering C++, Addison-Wesley, 1994.
9. Frogley, Thaddaeus, “An Introduction to C++ Traits,” http://thad.notagoth.org/cpptraits_intro
10. Gurtovoy, Aleksey and Abrahams, David, The Boost MPL Library, http://www.boost.org/libs/mpl/doc/index.html
11. ISO/IEC, International Standard: Programming Languages – C++, Second Edition, 15 October 2003, section 3.4.2 (“Argument-dependent name lookup”).
12. ISO/IEC, International Standard: Programming Languages – C++, Second Edition, 15 October 2003, section 10.3 (“Virtual functions”), paragraph 5.
13. Järvi, Jaakko et al., “Function Overloading Based on Arbitrary Properties of Types,” C/C++ Users Journal, June 2003, available at http://www.ddj.com/cpp/184401659.
14. Lippman, Stanley, Inside the C++ Object Model, Addison Wesley, 1996, pp. 95-101.
15. Meyers, Scott, “Using namespaces to partition code,” Usenet newsgroup comp.lang.c++.moderated, February 13, 2004, http://tinyurl.com/357n6w.
16. Meyers, Scott, Effective C++, Third Edition, Addison-Wesley, 2005, Item 29.
17. Meyers, Scott, Effective C++, Third Edition, Addison-Wesley, 2005, Item 47.
18. Meyers, Scott, “Red Code, Green Code: Generalizing const,” presentation to the Northwest C++ Users Group, April 2007. Video available at http://video.google.com/videoplay?docid=-4728145737208991310&hl=en, presentation materials at http://www.nwcpp.org/Downloads/2007/redcode_-_updated.pdf.
19. Ragle, Dan, “Introduction to Perl's Taint Mode,” http://www.webreference.com/programming/perl/taint/.
20. Sutter, Herb, “Thoughts on Scott's ‘Red Code / Green Code' Talk,” May 6, 2007, http://herbsutter.spaces.live.com/blog/cns!2D4327CC297151BB!207.entry.
21. Watanabe, Steven, “Re: [mpl] Hierarchy Generation,” Boost User's Mailing List, February 25, 2008.
22. Wikipedia, “Argument dependent name lookup,” http://en.wikipedia.org/wiki/Argument_dependent_name_lookup.
23. Zolman, Leor, “An STL Error Message Decryptor for Visual C++,” C/C++ Users Journal, July 2001, available at http://www.ddj.com/cpp/184401416.■

作者简介

Scott Meyers，C++顶级权威之一，为世界各地客户提供培训和咨询服务。著有Effective C++系列图书（《Effective C++》、《More Effective C++》和《Effective STL》），设计了创新型的Effective C++ CD，Addison Wesley的Effective Software Development Series顾问编辑，The C++ Source (http://www.artima.com/cppsource/)咨询板块专家。布朗大学计算机科学博士，他的信箱是smeyers@aristeia.com。

■ 责任编辑：赵健平（zhaojp@csdn.net）

基于Linux模块的防火墙系统

本文首先分析了Linux的Netfilter内核架构。并在此基础上，采用模块编程方式开发了一个高效实用的包过滤型防火墙系统。

■ 文 / 王占刚 王泽恒

引言

在计算机网络技术飞速发展的今天，网络安全问题一直困扰着人们。防火墙便是为了保证网络安全而架设的计算机硬件或者软件系统。商品化的防火墙产品，由于是通用软件，存在针对性不强的缺点。另外，在一些信息敏感场所，如政府和军事部门，对于无法了解其内部运行机制的安全产品一般不予选用。所以在很多场合，需要自行设计和开发防火墙系统。

近年来，Linux防火墙系统发展很快。这既得益于Linux的自由软件属性，也与其是一个高性能的网络操作系统密不可分。早在1995年，IPFWADM被植入Linux核心，Linux便具有了内置的防火墙功能。1999年，Linux的防火墙功能由Ipchains替代。Ipchains提供了规则链、碎片控制、NAT等新功能。但由于当时的Linux防火墙包括核心级和用户级代码，因此只要植入新的Linux防火墙代码，就必须改写相关的核心和用户空间代码。随着Linux2.4内核中Netfilter的出现，模块化的架构方式开始得到应用。模块能够嵌入运行的核心，以提供附加的通讯功能。

Netfilter框架

Netfilter是Linux2.4/2.6内核防火墙的实现框架，是迄今为止Linux内核中功能最完善强大的防火墙子系统。在内核中，通过Netfilter结构将防火墙对数据包的处理引入到IP层中实现。防火墙的代码与IP层的代码完全分离，从而构成不同的模块，使得防火墙的修改和扩充更加方便。Netfilter框架包含以下三部分：

（1）为每种网络协议(IPv4、IPv6等)定义一套钩子函数(IPv4定义了5个钩子函数)，这些钩子函数在数据报流过协议栈的几个关键点被调用。在这几个点上，协议栈以数据报和钩子函数标号作为参数调用Netfilter框架。

（2）内核的任何模块可以对每种协议的一个或多个钩子进行注册，实现挂接。这样当某个数据包被传递给Netfilter框架时，内核能检测是否有模块对该协议和钩子函数进行了注册。如果注册了，就调用该模块注册时使用的回调函数，这样该模块便有机会检查、修改甚至丢弃该数据包以及指示Netfilter将该数据包传入用户空间队列。

（3）排队的数据包传递给用户空间进行异步处理。用户进程可以检查、修改数据包，甚至可以通过一个钩子函数将数据包注入到内核中。

数据包通过Netfilter的流程如图1所示。

从图1中可以看到Netfilter共有5个钩子函数，分别为：

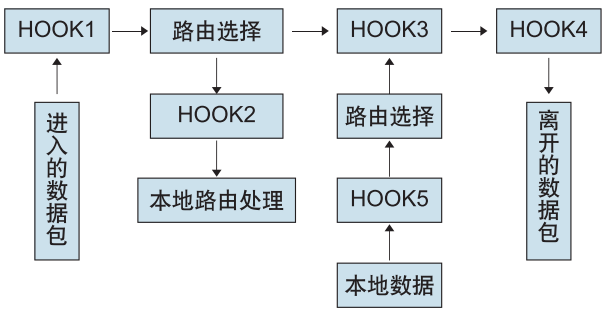


图1 数据包在Netfilter中的路径

- Hook1：NF_IP_PRE_ROUTING
- Hook2：NF_IP_LOCAL_IN
- Hook3：NF_IP_FORWARD
- Hook4：NF_IP_POST_ROUTING
- Hook5：NF_IP_LOCAL_OUT

数据报从左边进入系统，进行IP校验以后，数据报经过第一个钩子函数Hook1进行处理；然后就进入路由代码，其决定该数据包是需要转发还是发给本机的；若该数据包是发往本机的，则该数据经过钩子函数Hook2处理后传递给上层协议；若该数据包应该被转发则它被Hook3处理；经过转发的数据报经过最后一个钩子函Hook4处理以后，再传输到网络上。本地产生的数据包经过钩子函数Hook5处理后再进行路由选择处理，然后经过Hook4处理后发送到网络上。

内核模块可以对一个或多个这样的钩子函数进行注册挂接，并且在数据报经过这些钩子函数时被调用，从而模块可以修改这些数据报，并向Netfilter返回如下值：

- NF_ACCEPT　　继续正常传输数据报
- NF_DROP　　丢弃该数据报，不再传输
- NF_STOLEN　　模块接管该数据报，不要继续传输该数据报
- NF_REPEAT　　再次调用该钩子函数
- NF_QUEUE　　对数据报进行排队

注册一个hook函数是围绕nf_hook_ops数据结构的一个非常简单的操作，该数据结构在linux/netfilter.h中定义，具体如下：

```
struct nf_hook_ops {
    struct list_head list;
    nf_hookfn *hook;
    int pf;
    int hooknum;
    int priority;   };
```

其中，list用于维护Netfilter hook的列表；hook是一个指向nf_hookfn类型的函数的指针，该函数是这个hook被调用时执行的函数；pf用于指定协议族；hooknum用于指定安装的这个函数对应的具体的hook类型；priority用于指定在执行的顺序中，该hook函数应当在被放在什么地方。以下示例代码简单的注册了一个丢弃所有到达的数据包的函数，该代码同时演示了Netfilter的返回值是如何被解析的。

```
#define _KERNEL_
#define MODULE
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
static struct nf_hook_ops nfho;  /* 用于注册函数的数据结构 */
unsigned int hook_func(unsigned int hooknum,  /* 注册的hook函数的实现 */
                       struct sk_buff **skb,
                       const struct net_device *in,
                       const struct net_device *out,
                       int (*okfn)(struct sk_buff *))
{
    return NF_DROP;     /* 丢弃所有的数据包 */
}
int init_module()
{
    /* 填充hook数据结构 */
    nfho.hook = hook_func;                  /* 处理函数 */
    nfho.hooknum  = NF_IP_PRE_ROUTING;    /* 使用IPv4的第一个hook */
    nfho.pf       = PF_INET;
    nfho.priority = NF_IP_PRI_FIRST;      /* 让函数首先执行*/
    nf_register_hook(&nfho);              /*注册钩子函数*/
    return 0;
}
void cleanup_module()                     /* 清除程序 */
{
    nf_unregister_hook(&nfho);
}
```

系统功能概述

本防火墙系统基于Netfilter内核框架进行开发，可实现对数据包基于协议、IP地址、端口进行过滤。为了捕获进出的数据包，应支持根据用户的需求来设置检查规则。支持防火墙工作状态监控以及日志记录。可建立如图2所示的功能模型：

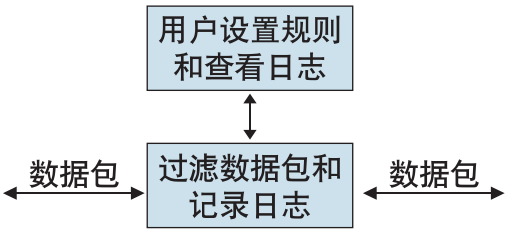


图2 系统功能模型

由于要对通过的全部数据包进行检查，然后再提交上层处理，防火墙本身的计算量必然较大。因此数据包过滤部分应该在系统内核中处理，以便实现高速运算和对原始数据包的准确控制。只有这样，才能实现安全与速度的合理平衡。

考虑到该系统的性能优先性，以及操作用户的专业性，一个方便快捷的命令行界面相比较图形界面更适合本系统。对于系统的开关控制以及规则设定，都通过该命令行界面实现。系统提供在线帮助。

系统设计

整体框架

系统整体可以分为三个部分来设计：用户界面部分，过滤功能部分和用户—过滤功能交互部分。用户界面部分直接在用户层实现。过滤部分则基于Netfilter体系架构，利用内核模块实现。用户—过滤交互部分承担着用户与过滤功能模块的命令和数据通信，沟通了用户层和内核层。考虑创建一个虚拟的设备文件，把过滤功能模块作为一个设备驱动器，而把用户界面部分作为应用程序，这样可以通过访问设备文件实现用户和内核模块的通信。

功能模块设计

为了对通过防火墙的数据包进行过滤，可以在Netfilter的5个hook点上设置检查点。对于进入的数据包，可以在Netfilter的Hook1（NF_IP_PRE_ROUTING）点上进行检查。因为Hook1是在数据包被校验之后，任何进入防火墙的数据包都会经过这里。而对发出包的检查可以设置在Hook4（NF_IP_POST_ROUTING）点上，这里是所有数据包流出防火墙的必经之处。

在这些检查点上，可以过滤协议，例如为了屏蔽ping探测；可以过滤进入包的源地址，以便阻止进入子网的关键机器；可以过滤进入包的目的地址；对于特定的服务也可以过滤端口号。同样对于离开包也可以进行以上的过滤。当然用户也可以动态地开启或关闭防火墙的过滤功能。

再接下来就是设定规则。可以将规则组织成链，链中只要有一个规则不符合，数据包就会被拒绝。规则链可以组织成两条：进入规则链和离开规则链。

在处理数据之后，需要记录防火墙处理数据包的方式以及数量，为流量分析和异常控制做好准备。还需要设计一个显示状态的功能的模块，以便用户查看防火墙的工作状态。系统主要的功能模块构成如图3所示。

交互设计

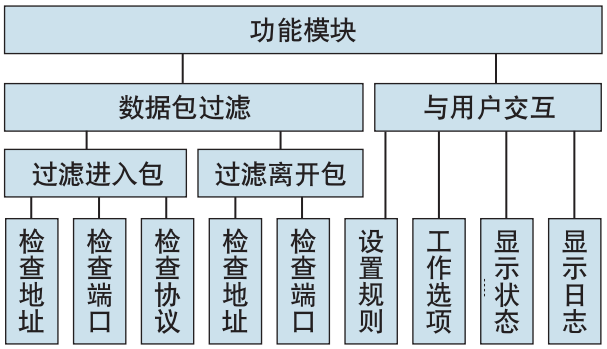


图3 防火墙功能模块设计

这一部分联系着界面和内核功能模块两部分，它几乎是虚拟的，但又是必不可少的。必须在此定义一些公共的数据结构，只有这样用户部分和内核功能模块部分才能顺利地交换数据。其设计如图4所示。

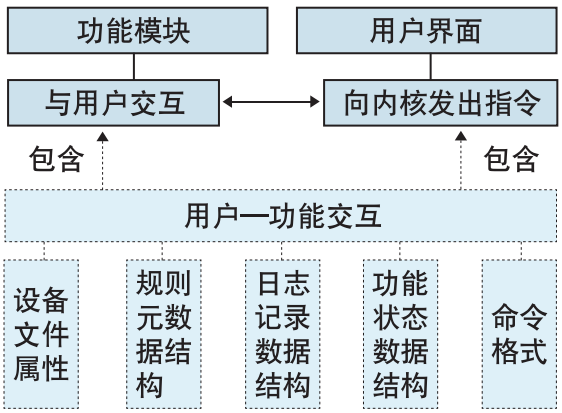


图4 交互设计

系统实现

该系统在RedHat9.0上实现，内核版本号为2.4.20。编译运行后，便可以测试与用户的交互以及命令的执行。运行及测试结果如图5所示。

图5 防火墙系统运行界面

结论

通过以上分析，证明基于Linux的防火墙系统开发是完全可行的，且完成的系统稳定高效。由于Linux的开源性，即使用户不能亲自开发防火墙系统，也可以通过修改别人的系统使之适应个性化应用。所以说，Linux下的防火墙实现机制是一个非常值得深入探讨的课题。

参考文献

[1] 胡建理，王嘉祯，刘爱珍著.基于Linux2.4内核的防火墙技术及应用研究[J].微电子学与计算机,2006,(4):168-171.

[2] 刘建峰，潘军，李祥和.Linux防火墙内核中Netfilter和Iptables的分析[J].微计算机信息,2006,(1):7-9.

[3] Peter Jay Salzman. The Linux Kernel Module Programming Guide. http://www.faqs.org/docs/kernel/, 2003-04-04.

[4] 倪继利.Linux内核分析及编程[M].北京：电子工业出版社,2005.

[5] 王一平，韦卫.网络安全框架Netfilter在Linux中的实现[J].计算机工程与设计,2006,(3):439-442.

[6] 毛新宇.Linux内核防火墙Netfilter的原理和应用[J].微型机与应用2004,(4):35-37.

作者简介

王占刚，讲师，博士在读，研究方向：网络安全、软件测试。

王泽恒，工程师，研究方向：应用数学。

Linux网卡驱动分析一例

■ 文 / 独孤求真

概述

对于内核驱动程序的初学者来说，常常遇到的问题就是搞不明白一个驱动程序是如何驱动硬件设备工作的，这主要是由于这方面的资料比较少，而现有的资料对一些关键的基础点或是避而不谈，或是故作神秘一笔带过。本文将以8139网卡驱动程序为例，介绍一个设备驱动程序究竟是如何驱动硬件设备工作的，同时对相关的基础知识进行介绍。

总线驱动程序与物理设备对象

在系统启动阶段，PCI总线驱动程序根据PCI总线协议进行总线扫描，并且为每一个设备建立一个物理设备对象pci dev，该结构的主要成员如图1所示：

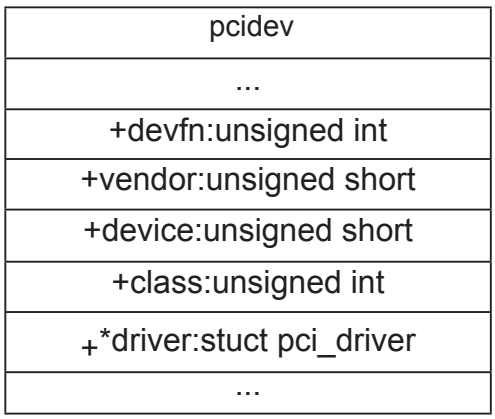


图1 PCI物理设备对象

读者可以从Linux内核中的pci.h看到pci_dev结构的详细定义，这里的devfn，就是从PCI配置空间读取出来的设备号和功能号，同样vendor为厂商号，等等。而最后的driver指向一个驱动程序对象，PCI的设备驱动程序对象定义为pci_driver。

设备驱动程序对象及其注册

总线驱动程序为总线上的每一个PCI设备建立了物理设备对象pci dev，之后就要让各个设备驱动程序来认领自己能驱动的设备了。我们来看看8139的设备驱动程序对象的定义：

```
static struct pci_driver rtl8139_pci_driver = {
       .name       = DRV_NAME,
       .id_table   = rtl8139_pci_tbl,
       .probe      = rtl8139_init_one,
       .remove     = __devexit_p(rtl8139_remove_on
e),
#ifdef CONFIG_PM
       .suspend    = rtl8139_suspend,
       .resume     = rtl8139_resume,
#endif /* CONFIG_PM */
};
```

代码1 PCI驱动程序对象

在这里最重要的成员结构就是id_table和probe，其中id_table指定了这个驱动程序能驱动哪些设备。rtl8139的pci tbl定义如下：

```
static struct pci_device_id rtl8139_pci_tbl[] = {
        {0x10ec, 0x8139, PCI_ANY_ID, PCI_ANY_ID,
0, 0, RTL8139 },
        {0x10ec, 0x8138, PCI_ANY_ID, PCI_ANY_ID,
0, 0, RTL8139 },
        {0x1113, 0x1211, PCI_ANY_ID, PCI_ANY_ID,
0, 0, RTL8139 },
        ......
}
```

代码2 rtl8139的id_table

这个驱动程序在加载时调用pci register driver()注册了一个驱动程序对象：

```
static int __init rtl8139_init_module (void)
{
         return pci_register_driver(&rtl8139_pci_
driver);
}
```

代码3 驱动程序对象的注册

前面说过物理设备对象中保存着设备id，设备产商id，类别id等信息，而这些信息是从设备上读取出来的，它对应一个实际的物理设备，而设备驱动程序对象的id table也有一份这样的信息，它表示这个驱动程序能够“驱动”哪些设备。有了这些信息，现在系统就可以建为一个物理设备找到它的驱动程序对象了，之后就会调用驱动程序对象中的probe函数，这个函数的作用就是接管一个物理设备对象，并且完成必要的初始化工作。

设备初始化

系统为物理设备对象找到对应的驱动程序对象后，就会调用它的初始化函数，其中一个参数就是物理设备对象指针。8139的初始化函数为rtl8139_init_one().

```
static int __devinit rtl8139_init_one (struct
pci_dev *pdev,
                              const struct
pci_device_id *ent)
{
       struct net_device *dev = NULL;
       struct rtl8139_private *tp;
       int i, addr_len, option;
       ......

i = rtl8139_init_board (pdev, &dev);
if (i < 0)
                return i;

       tp = netdev priv(dev);
tp->dev = dev;
io_addr = tp->mmio_addr;

dev->open = rtl8139_open;
dev->hard_start_xmit = rtl8139_start_xmit;
netif_napi_add(dev, &tp->napi, rtl8139_poll, 64);
dev->stop = rtl8139_close;
dev->get_stats = rtl8139_get_stats;
dev->set_multicast_list = rtl8139_set_rx_mode;
dev->do_ioctl = netdev_ioctl;
dev->ethtool_ops = &rtl8139_ethtool_ops;
dev->tx_timeout = rtl8139_tx_timeout;
dev->watchdog_timeo = TX_TIMEOUT;
......
i = register_netdev (dev);
       if (i) goto err_out;
......
}
```

代码4 设备初始化函数

在上面的代码中，首先调用rtl8139_init_board()对网卡设备进行初始化，同时这个函数会调用alloc_etherdev()分配一个net_device对象，我们把这个设备对象称为功能设备对象，此后调用register_netdev()注册一个功能设备。注册成功后，在shell中运行ifconfig -a就可以看到这个网卡了。

这以后，通过tp->mmio_addr获取了设备的基地址，这里tp->mmio_addr是rtl8139_init_board()根据配置空间中的基地址寄存器值建立起来的虚拟地址映射，以后CPU以这个地址加上表1中的偏移，就可以访问到设备上对应的寄存器了。

在这里，我们需要弄清楚几种常见的地址的区别：

- 虚拟地址

在保护模式下使用的地址，开启了分页机制后，CPU只能使用虚拟地址进行寻址访问，这个地址经过CPU的MMU转换得到物理地址，也就是通过页表，页目录的映射转换，然后传递到北桥。

- 物理地址

在实模式或者经过CPU的MMU转换后得到的地址。

- 总线地址

经过各种桥接器转换后，出现在各总线上的地址，例如：HostBridge, PCI-PCI Bridge.在配置期间，CPU写入配置空间中Base Address Register的是总线地址。通常总线地址等于物理地址，但是也可能不相等，这取决于系统的设计。而开启分页机制后，CPU访问的必须是虚拟地址，因此在初始化函数中，需要通过页表，建立物理地址到虚拟地址的映射。而如果总线地址和物理地址不相等，就需要根据总线地址和物理地址的映射机制计算出它的物理地址，然后进一步建立物理地址到虚拟地址的映射。

在net device结构中，最重要的就是上面设置的一组操作函数，这些函数说明如表1所示。

函数指针	说明
Open	打开设备时调用，例如在shell中运行ifconfigethn up，最终会调用该个函数
Stop	关闭设备时调用，例如在shell中运行ifconfigethn down，会调用该函数
get_stats	获取统计信息，例如在shell中运行ifconfig ethn，会看到收发了多少个数据包等统计信息，其数据就是调用这这个函数获取的
do_ioctl	设备控制函数，例如对设备调用ioctl函数，将调用这个函数
Ethtool_ops	ethtool的控制接口，例如在shell中运行ethtool ethn，会调用这个函数
…	…

表1 网卡设备的初始化

现在驱动程序以及完成了对物理设备对象的认领工作，并且已经注册了相应的功能设备对象。这样就可以为上层应用提供某项具体的功能了，大家都知道，网卡提供的功能就是进行网络数据的接收和发送，这个工作又是如何完成的呢？下面我们来看看是如何启用网卡设备的。

网卡设备的初始化

功能设备的初始化工作由rtl8139_open()这个函数完

成的，这个函数需要进一步申请资源，并且启用网卡。在了解初始化工作之前，有必要先介绍一下发送和接收的大致过程，网卡内部有软件不可见的存储空间，网卡把网线上传输的数据保存到这个空间，同时会通过DMA操作把数据传送到内存，而这个内存地址由接收地址寄存器指定。因此在接收方面，rtl8139_open()必须分配一块内存，然后把这个内存的首地址写入到接收地址寄存器中。网卡把数据传送到内存区域后，会发出中断通知CPU，之后CPU可以把这部分数据上交到协议层进行进一步的处理。当传递到这片内存的末尾时，网卡又从内存的最前面开始传递，因此这是一个环形缓冲区，如果CPU不计时处理，数据包有可能会被覆盖。这里需要注意的是，由于外部设备没有MMU单元，而CPU分配内存使用的是虚拟地址，写入接收地址寄存器的地址必须为物理地址(或者总线地址，如果两者不相等的话。)，另外地址必须是物理连续的。

在发送方面，8139网卡有4个发送地址寄存器，CPU把待发送的数据写入到这4个寄存器中的任何一个，然后向命令寄存器中写入一条发送命令，网卡就开始进行发送了，发送结束后，网卡会发出中断通知CPU。由于CPU的速度远远快于外设的速度，因此轮流使用这四组发送寄存器。最后需要注意提醒的是各个网卡的发送，接收的组织方式差别很大，但是原理都一样，在这里彻底搞清楚每一个寄存器的每个bit的意义，是没有必要的。

现在我们来看看open()操作：

```
static int rtl8139_open (struct net_device *dev)
{
......
retval = request_irq (dev->irq, rtl8139_interrupt,
IRQF_SHARED, dev->name, dev);
if (retval)
return retval;

tp->tx_bufs = dma_alloc_coherent(&tp-
>pci_dev->dev,
TX_BUF_TOT_LEN,
&tp->tx_bufs_dma,
GFP_KERNEL);

tp->rx_ring = dma_alloc_coherent(&tp-
>pci_dev->dev,
RX_BUF_TOT_LEN,
&tp->rx_ring_dma,
GFP_KERNEL);
rtl8139_init_ring (dev);
rtl8139_hw_start (dev);
......
retrun 0;
}
```

代码5 rtl8139_open

这里主要的工作就是注册中断处理函数，分配发送和接收缓冲区，然后对这些缓存区及其管理结构进行初始化，再把相关的信息写到网卡上的对应的寄存器中。

发送处理

```
static int rtl8139_start_xmit (struct sk_buff
*skb,
struct net_device *dev)
{
entry = tp->cur_tx % NUM_TX_DESC;
skb_copy_and_csum_dev(skb, tp->tx_buf[entry]);

spin_lock_irqsave(&tp->lock, flags);
RTL_W32_F (TxStatus0 + (entry * sizeof (u32)),
tp->tx_flag | max(len, (unsigned int)ETH_ZLEN));

}
```

代码6 rtl8139_start_xmit

这个函数的第一个参数skb，就是待发送的数据包，首先计算出要使用哪一个发送地址寄存器，然后把数据拷贝到相应的缓存区中，再把这个地址，以及待发送数据包的长度等信息写入相关寄存器，现在网卡就开始发送数据了。当发送完成后，网卡发出中断，CPU会调用前面注册的中断处理函数rtl8139_interrupt()。

中断处理

中断处理的主要工作就是读取设备上的中断状态寄存器，根据状态判断是发送完成中断，接收完成中断，或是出错中断等等，然后根据不同的情况做不同的处理。

```
static irqreturn_t rtl8139_interrupt (int irq,
void *dev_instance)
{
      ......
spin_lock (&tp->lock);
status = RTL_R16 (IntrStatus);
......
/* 接收中断。*/
if (status & RxAckBits){
if (netif_rx_schedule_prep(dev, &tp->napi)) {
RTL_W16_F (IntrMask, rtl8139_norx_intr_mask);
__netif_rx_schedule(dev, &tp->napi);
}
}

      /* 错误中断。*/
      if (unlikely(status & (PCIErr | PCSTimeout
| RxUnderrun | RxErr)))
rtl8139_weird_interrupt (dev, tp, ioaddr,status,
link_changed);

if (status & (TxOK | TxErr)) {
rtl8139_tx_interrupt (dev, tp, ioaddr);
if (status & TxErr)
```

```
RTL_W16 (IntrStatus, TxErr);
}
……
}
```

代码7 rtl8139_interrupt

从这里可以看出，在中断处理中，最重要的工作就是根据状态寄存器的值判断产生中断的原因，然后不同的中断处理函数。

接收处理

在中断处理中我们已经看到，对于接收中断会调用netif_rx_schedule()，这个函数会调度一个软中断，这里涉及到软中断和NAPI的东西，由于篇幅有限，我们不再进行讨论。最终，在软中断环境下，会调用rtl8139_rx()这个函数进行接收处理：

```
static int rtl8139_rx(struct net_device *dev,
struct rtl8139_private *tp,
int budget)
{
        ……
        while (…) {
                ……
                skb_copy_to_linear_data (skb,
&rx_ring[ring_offset + 4], pkt_size);
                skb_put (skb, pkt_size);
                netif_receive_skb (skb);
```

```
}
return received;
}
```

代码8 rtl8139_rx

这个函数的主要工作就是分配一个skb结构，再把数据包从接收缓冲区中拷贝到skb中，最后把这个数据包传递到协议层，接下来协议驱动就可以接手处理这个包。由于篇幅关系，对网卡驱动程序的结束就到此为止了，文章中省略了很多细节，最后在这里为感兴趣的读者介绍相关的参考资料：

- 关于PCI总线协议请参考《PCI Local Bus Specification》及相关文档。
- 关于软件如何实现PCI总线扫描，配置等信息请参考《Linux内核源代码情景分析》及阅读内核相关部分的代码。
- 关于8139网卡的细节请参考8139网卡的Datasheet以及Programming Guide. ■

作者简介

李云华，软件工程师。长期从事操作系统内核、驱动程序，以及嵌入式系统方面的开发和研究。

Open API分析与实践

Open API是近一年来比较流行的互联网技术，本文作者结合自己的经验，讨论了Open API的理念和实践两方面问题。

■ 文 / 岑文初

分析篇

Open API的形态

就现在互联网上Open API的形态来看，主要分成两种：标准REST和类REST(也可以叫做RPC形态)。RPC形态其实就是Web Service的一种延续，只是少了繁重的解析、安全规范等。Flickr的Open API大部分就是这种形态，看看下面的服务请求URL：

http://api.flickr.com/services/rest/?method=flickr.test.echo&name=value

服务请求地址包括了两部分：1.服务的总入口地址http://api.flickr.com/services/rest/。2.服务方法以及参数。这和过去的RPC模式就是一样的，只是通过Http方式请求，返回的是可以指定格式数据内容。

REST形态主要有这么几点特点：1.服务地址就是资源定位地址。2.服务操作就是Http请求中的方法类型（GET,POST,DELETE,PUT），这其实是抽象现实当中对于服务的增删改查操作。Google大部分的REST API就采用了标准的REST风格，服务请求地址URL如下：

http://www.google.com/calendar/feeds/wenchu.cenwc@alibaba-inc.com/allcalendars/full

这个服务请求地址是用来定位我阿里巴巴邮箱注册的Google帐号的所有日程安排，通过在Http消息头中配置Get、Post、DELETE、PUT可以对我的日程进行操作，而无须登陆到Google上去操作。（后面部分的实践中会有部分介绍如何通过后台Java代码直接操作）

对于REST形式的讨论在网上一直有，但这种讨论没有什么意义，其实就好比争论吃饭是否一定要用筷子，没有什么技术是“万能药”，也没有什么技术好与不好，只有使用它的人是否有足够的智慧把它应用到适合的场景中。

对于类REST的形态来说，优点在于对于原有系统的改造较小，“当前”用户使用接受度更高一些，对于逻辑抽象来说更加容易。而REST风格的优点在于，资源容易管理，系统扩展容易，权限控制可以部分依托于已有的传输协议。两者的缺点其实就是对方的优点。采取什么模式，还是要根据企业本身情况来看，淘宝采用的就是类REST方式，而未来支付宝将会采用REST的方式。对于前者来说，要改造整个系统架构和资源数据结构基本是不太可能完成的任务，后者对于业务逻辑梳理以及在现有内部SOA架构体系下抽象出REST风格的API不是一件难事。但最后还是那句话，形态仅仅只是外在，练功之人修炼好内力才是根本，没有必要为了迎合潮流而去盲目的选择形态，因为服务提供商将要面对的是高过网站更高流量的访问调用，如何满足开发者业务以及非业务（稳定、高效、安全）的需求，才是最大的挑战。

Open API的类型

这里指的类型，主要从提供服务本身内容来看。当前服务类型主要可以分成三种：数据型，应用型，资源型。

现在很多SNS网站的Open API就是属于数据型，也就是将自身的数据开放，让应用开发者根据已有的数据进行二次应用开发。

应用型其实应该是数据型结合的比较紧密，Flickr的图片搜索，Google的日程，地图（地图数据其实可以自己定义）等等都是属于应用型。应用型的数据输入可以是外部的数据，也可以是基于已有的数据资源进行处理。

资源型的代表就是Amazon，Amazon S3就是典型的资源型，当然Flickr的图片存储服务等也可以属于资源型。其实还有一个被炒得火热的话题就是云计算，在云计算的背后就是需要提供这么一个资源型的服务，Amazon EC2如果离开了S3，也就无法存在。这种类型的服务也是一种未来的趋势，未来互联网应用如果要培植草根级开发者，就需要有这样的温室，Google 的App engine如果再多一些语言环境版本，那么会让更多的开发者有梦想实现的空间。

在回过头来看三种类型的服务，其实有着很强的层次关系，就如下图：

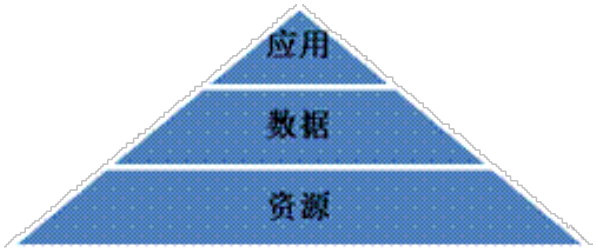


图 1 Open API的类型

Open API交互的数据格式

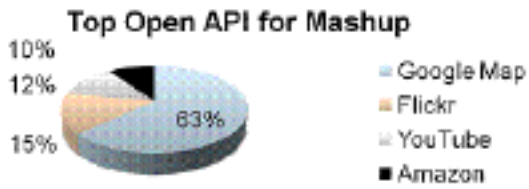
对于互联网应用来说，最大的特点也是最大的优点就是基于Http协议开发成为应用开发的统一标准。对于使用的语言，采用的操作系统和应用部署平台都没有太多的限制。Web Service采用xml作为数据传输承载，制定了解析标准（以及后来安全，转发等标准）为开发者异构系统的信息交互带来了可能，也成为至今为止应用最广泛的服务集成方式。而随着Web2.0发展，RSS、Atom、JSON的大规模应用，数据交互格式有了更多的选择。

服务请求就是标准的Http的请求，对于文件类上传的服务采用HTTP Multipart的格式，编码方式基本都采用UTF-8的编码方式。

在Open API的数据返回格式方面，大部分的网站优先提供Xml、JSON的数据返回，Google定义的GData就是在Atom基础上作了扩展，还有一些网站提供了PHP的数据返回。同时有些网站会在Open API的基础上作更高的一层封装，类似于Google Map，可以通过JavaScript框架来直接使用。

当前国外Open API使用状况

我这里只列了前四名的一个比例对照，但是前面四名占有总的Mash-up 的比例已经高达80%左右。



从这个占有率可以看出，API首先吸引开发者的应该是API应用场景是否广泛，Google Map其实就是最好的说明，地图类服务可以和各种行业结合起来为人们生活服务。其次就是API的专业化，后面三位这方面都是本类服务中做的最出色或者说是暂时还没有人可以做到的。Flickr的服务就是围绕着图片，但是Flickr对于图片Tag专业性的设计让使用者的需求得到了最大的满足，同时也为开发者提供了很多隐性的资源。YouTube借助着Google在搜索领域的强大优势以及自身的行业能力也吸引了广大的开发者。而Amazon多层次的API结构化设计，为开发者提供了整套的开发解决方案（EC2,S3,SQS,SimpleDB作为基础的Framework; FPS,DevPay作为配套支付服务支持，Alexa Web Search作为搜索），同时加上自身的强大的电子商务基础，也成为了很多开发者的首选。

从国外的Open API来看，如果要成为开发者首选，需要在服务特色，服务质量，服务配套化（社区，SDK，开发框架，整体解决方案）上做文章。很多企业已经有了吸引人的数据资源（类似于淘宝，YouTube，Flickr），或者拥有行业内强大的专业能力（类似于Google的搜索，地图，支付宝的支付）都可以比较容易地占有市场优势，而类似的，国内很多SNS网站商业模式已经被复制的差不多了，数据内容不分上下，因此如何能够做好服务特色、质量、配套化才是未来在Open API领域走的更远的基石。

实践篇

Open API实践部分根据授权策略的不同和使用方式的不同分成几阶段内容，完整的代码可以去Google Code下载（http://rest-demo.googlecode.com/files/demostore.rar）。注：代码中的部分用户名和密码以及应用id都需要采用自己申请的内容作替换，代码都是Java的后台程序代码，主要考虑实践即可，同时这部分代码仅仅是为了测试，结构和错误处理都是没有做太多的关注。

三类授权策略

分类	类型	访问对象	作用	安全级别	手段
免授权		公开类信息（例如搜索引擎搜索到的结果等）		低	申请应用ID
应用授权	非商业商业	用户相关公开类信息（例如SNS中的个人公开信息）	服务提供商交验应用身份，保证数据传输无篡改	中	数字签名
用户授权	Web模式桌面模式手机模式	个人用户相关的非公开信息（例如淘宝用户隐私信息）	在保障用户个人隐私数据的安全性前提下，提供给应用开发者访问和操作用户个人信息的能力	高	1.OAuth代理授权 2.用户名、密码代理授权

免授权只需要开发者申请应用ID即可使用服务。应用授权是最基础的Open API开发授权策略，作用是让服务提供商能够核对每一次服务请求者的身份，同时也保证了服务开发商的自身利益，与免授权之间的区别就是是否需要在请求中带上数字签名来交验请求者身份。用户授权

一般是基于应用授权之上的更高层次的授权认证，为了保障终端用户数据不会在用户不知情或未授权的情况下被访问和修改，造成用户隐私泄露或者蒙受损失。

免授权和应用授权类服务的开发

Yahoo 的 Search 引擎以及 Boss 服务（Build your Search Service）都是属于免授权类服务。

首先，上 Yahoo 开发者网站申请应用身份（http://developer.yahoo.com/），这里首先需要拥有一个 Yahoo 的 Id，然后即可申请应用 Id，一个开发者可以申请多个应用 Id。

客户端测试代码片段如下：

```
        @Test
        public void testYahooSearch()//Yahoo搜索服务测试
        {
                String yahoo_search_service = "http://search.yahooapis.com/{searchengine}/{version}/{searchtype}";//Yahoo搜索服务地址，通过切换searchengine,searchtype来改变搜索资源的不同，支持图片，网页，新闻，视频
                Map<String,Object> params = new HashMap<String,Object>();//服务调用传入的参数
                params.put("appid", yahoo_appkey);//申请的应用身份Id
                params.put("query","冰激凌 +巧克力 -水果");//搜索内容，搜索冰激凌，需要有巧克力但不要水果
                //ImageSearchService，图片类搜索,TestUtil是一个基础工具方法，这里就不贴了，详细地可以参看Google code的源码
                String result = TestUtil.sendRequest(yahoo_search_service
                                                     .replace("{searchengine}", "ImageSearchService").replace("{version}", "V1").replace("{searchtype}", "imageSearch")

,params,null,TestUtil.HTTP_METHOD_GET,null,"UTF-8",null);
                System.out.println(result);
                ……
        }

        //这部分将在后面的Mash-up范例中使用，是通过Boss搜索缩略图片
        public static List<String> bossSearchService() throws XPathExpressionException, ParserConfigurationException, SAXException, IOException
        {
                ……
                String boss_image_service = "http://boss.yahooapis.com/ysearch/images/v1/";//服务地址
                String resource = URLEncoder.encode("\"ice cream\" -apple +Chocolate","UTF-8");//搜索资源关键字编码，关于编码特别要注意，很多时候使用Open API出现问题就是因为中文，导致签名或者使用出现问题

                ……
                String result = TestUtil.sendRequest(new StringBuffer().append(boss_image_service).append(resource).toString()

,params,null,TestUtil.HTTP_METHOD_GET,null,"UTF-8",null);
                urls = processXmlResult(result);//采用XPath来解析返回结果，当前通常解析返回结果的方式就采用XPath,XQuery，如果是RSS方式通常会采用一些第三方的开源项目，类似于ROME，详细内容可以参看Google code的源码
                return urls;
        }
```

接下来就看看运行效果吧：

```
testYahooSearch()的运行结果如下：
<?xml version="1.0" encoding="UTF-8"?>
<ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:yahoo:srchmi" xsi:schemaLocation="urn:yahoo:srchmi http://api.search.yahoo.com/ImageSearchService/V1/ImageSearchResponse.xsd" totalResultsAvailable="171" totalResultsReturned="10" firstResultPosition="1">//记录数信息
<Result>
<Title>1194498494726_77635.jpg</Title>
<Summary>]冰激凌2.5万美元1份 ……</Summary>
<Url>http://news.tom.com/uimg/2007/11/8/chenhong/1194498494726_77635.jpg</Url>
  ……
</Result>
<Result>……</Result>
……
</ResultSet>
```

测试运行结果是搜索结果集的 xml 描述，可以根据 ImageSearchResponse.xsd 来解析返回的内容。

testBossSearch()运行的结果如下：

```
nextpage Url :
/ysearch/images/v1/%22ice%20cream%22%20-apple%20%20Chocolate?count=10&dimensions=small&appid=nkl8kwzV34FPuapz_cGP3QiOU7jvOZB2kuWEBq0CoGBvRfirCtgnIMP6mVYNHRuFWBHn&format=xml&start=10
search result 1 url : http://www.soya.be/pictures/market/chocolate-ice-cream.jpg
search result 2 url : http://www.icecreampark.com/images/Chocolate%20ice%20cream%202.jpg
……
```

测试运行的结果是已经经过 XPath 初步处理的结果，提供了下一页的入口 URL 地址，以及本次搜索出来的结果集。

通过阿里软件服务集成平台访问淘宝非用户隐私信息类 API 就属于应用授权类服务。与上面范例差异在于调用发送方法时传入了 secretcode，进行参数签名（参数中增

加了时间戳）。

TestUtil.sendRequest(url,params,secretcode,TestUtil.HTTP_METHOD_POST,null,"UTF-8",null);

由上面的例子可以看出，对于公开信息的访问，Open API接入简单，使用方便。

用户授权类服务的开发

用户授权类服务，首先要解决如何让用户能够在知情的情况下授权给应用开发商获取和操作用户个人的数据，实现用户需求。在传统意义上通常会让用户输入某一个网站的用户名和密码，就类似于现在的很多SNS让用户输入msn,qq帐号来获取用户好友信息，但是其实这样对于用户来说风险很大，特别是一些个人隐私性很强的信息或者是涉及到金钱的操作。因此现在大部分的服务提供商采取的是OAuth方式的认证或者是类似于OAuth，OAuth的具体细节我就不在这里赘述了，网上有很详细地资料，这

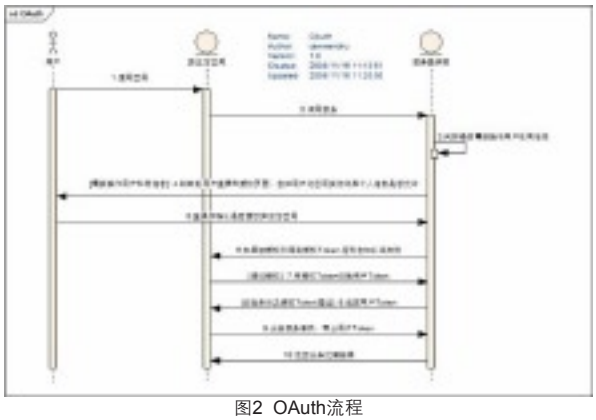


图2 OAuth流程

里就大致把流程原理画一下。

这里将采用Flickr图片上传作为测试范例：

首先，还是要拥有Flickr的帐号，然后同时去申请应用id。

具体的代码片断如下：

```
public class FilckrServiceTest
{
      ……
@SuppressWarnings("unchecked")//认证和授权流程，后面正式测试代码中会使用
      protected String auth() throws IOException
      {
            ……
            logicmap.put("method", "flickr.auth.getFrob");//授权服务名称作为参数
            logicmap.put("api_key", flickr_appkey); //置入应用ID
            //发起服务请求，获取授权Token
            result = flickrCommonServiceRequest(TestUtil.HTTP_METHOD_GET,flickr_service,logicmap,null);
            //分析返回结果获取授权Token
            frob = TestUtil.getTagContent(result,"<frob>","</frob>");

            if (frob != null)
            {
                  ……
                  //根据服务地址，apikey,授权Token,操作权限和参数签名，拼装用户授权URL
                  ……
                  //模拟弹出IE，展现给用户授权登录的界面
                  String cmd =new StringBuilder("rundll32   url.dll,FileProtocolHandler ").append(authStr).toString();
                  Runtime.getRuntime().exec(cmd);
                  ……
                  //当用户在弹出IE中授权完成以后，需要在控制台中输入ok回车表示确认，程序才能继续
                  String passcommand = TestUtil.readLineFromConsole();

                  if (passcommand.equals("ok"))//继续将授权Token交换成为用户Token
                  {
                        ……
                        result = flickrCommonServiceRequest(TestUtil.HTTP_METHOD_GET,
flickr_service,logicmap,null);
                        token = TestUtil.getTagContent(result,"<token>","</token>");//获得用户Token
                  }
            }
            return token;
      }

      @Test//测试上传图片
      public void uploadPhotoTest() throws IOException
      {
            String token = auth();//授权
            if (token != null)
            {
                  //访问个人上传图片各种状态信息，可以不作，这里只是确认还有多少图片可以上传
                  //logicmap中放入请求的方法名，apiKey,token
                  String result = flickrCommonServiceRequest(TestUtil.HTTP_METHOD_GET,
                                    flickr_service,logicmap,null);
                  ……

                  //输入需要上传图片的地址，可以是本地的图片也可以是网络图片
                  System.out.println("please input photo location you want to upload!");
                  String file = TestUtil.readLineFromConsole();
                  ……
                  //logicmap中是必要的业务参数，files是需要上传的文件地址
                  result = flickrCommonServiceRequest(TestUtil.HTTP_METHOD_POST,
                                    flickr_uploadphoto_service,logicmap,files);
                  //如果上传成功将会获得图片id
```

```
                        String photoid = TestUtil.
getTagContent(result,"<photoid>","</photoid>");

//根据返回的图片id拼装成资源访问地址，Flickr对于图片内容
的定位还是采用标准的REST风格，地址如下：
                http://www.flickr.com/photos/用户名
/photoid
                ……
}
```

看看执行效果：

首先控制台会输出：if done then input 'ok' to console!，同时弹出IE窗口如下：

输入用户名和密码以后会看到如下界面，就表示授权成功了。

在控制台中输入ok，然后回车。看到如下提示：

//这部分就是上传信息查询的结果

```
……
<user id="31642801@N06" ispro="0">
      <username>wenchu_cenwc</username>
      <bandwidth max="104857600" used="43078"
maxbytes="104857600" usedbytes="43078" remaini
ngbytes="104814522" maxkb="102400" usedkb="42"
remainingkb="102358" unlimited="0" />
      <filesize max="10485760" maxbytes="10485760"
maxkb="10240" maxmb="10" />
      <sets created="1" remaining="2" />
</user>
……
```

然后就输入你需要上传的图片的地址：

例如我输入我的blog的头像地址：http://avatar.profile.csdn.net/3/5/1/1_cenwenchu79.jpg

然后回车，会看到新弹出一个页面，里面就是上传到你Flickr中的图片。

以上就是一次用户授权API的完整操作，对比应用授权，个人授权相对来说会比较复杂一些，同时根据调用应用的不同，也会有不同的授权流程（Web应用，桌面应用，手机应用）。但就现在国内外的Open API使用来看，大致的思想都比较相似，也就是OAuth的思想，但是细节部分会有不少差异，例如Token时效，维护方，操作范围等。

Mash-up范例

Mash-up在基维百科中定义是这样的（In web development, a mash-up is a web application that combines data from more than one source into a single integrated tool）。数据的一种集成。Open API真正的目的就是希望够让信息在交流中产生更大的价值。

这个范例的场景是淘宝卖家上传上品信息的同时需要有商品的图片，通常商家就不得不自己再去找一些符合自己商品的缩略图，这里我采用上面使用过的Yahoo BOSS搜索缩略图，将符合条件的缩略图选择一个作为商品的描述图片再上传到淘宝。这样就将整个淘宝卖家编辑上传宝贝的流程简化了，并且对于商品图片描述来说会有更多更好的选择。

淘宝的Open API都通过阿里软件的服务集成平台发布，具体的使用流程其实和前面描述的两种服务获取方式一样，只是在用户授权方面对于应用开发者来说更加简便。

第一步，还是申请应用帐号和安全密码，不过需要首先拥有一个阿里巴巴中文站或者淘宝、阿里软件的帐号，然后登录http://isv.alisoft.com/isv/portal/home/home.jspa，然后选择开发者联盟的标签，里面有说明文档和指南。

第二步，你需要有一个淘宝帐号，并且开了网店。

第三步，客户端代码，代码片断如下：

```
public class TaoBaoServiceTest
{
       @Test
       public void updateProduct() throws
IOException, XPathExpressionException, ParserConfi
gurationException, SAXException
       {
                //模拟弹出IE窗口，通过服务集成平台根据服
务名称转入到相应的服务提供商的用户授权页面
                Runtime.getRuntime().exec("rundll32
url.dll,FileProtocolHandler   http://sip.alisoft.
com/sip/login?sip_apiname=taobao.item.update&sip_
appkey=13201&sip_sessionid=taobaotest1234567");
                System.out.println("if done then
input 'ok' to console!");
                //当登录完成并授权，则继续业务处理
                String passcommand = TestUtil.
readLineFromConsole();
                if (passcommand.equals("ok"))
                {
                        String url = "http://sip.
alisoft.com/sip/rest";//阿里软件服务集成平台的入口地址
                        String secretcode = "aa5b6
c3064fb11ddb63df45551ff8814";//应用签名私钥
                        String appkey = "xxx";//应
用ID

//先获取当前在卖的宝贝列表，构建服务请求参数，需要
appkey,api的名称,时间戳,sessionid，其中sessionId与上
面授权页面请求中的SessionId保持一致，用于平台转化为用户
Token。
                        ……
String result = TestUtil.sendRequest(url,params,
secretcode,TestUtil.HTTP_METHOD_POST,null,"UTF-
8",null);
                        //获取其中一个商品的id用于后面
信息上传
                        String iid = TestUtil.
getTagContent(result,"<iid>","</iid>");

                        if (iid != null && iid.
length() > 0)
                        {
//构建更新宝贝的服务请求参数，需要宝贝id,appkey,api的名
称,时间戳,sessionid，以及其他的淘宝服务相关的业务参数
                                …….
                                //通过Yahoo BOSS搜
索图片
                                ……
                                List<String> urls
= YahooServiceTest.bossSearchService();
                                if(urls.size() >
1)
                                {
                                        files.
put("image", urls.get(1).substring(urls.get(1).
indexOf("http://")));
                                }
                                else
                                        ……
                        //将图片和其他信息上传到淘宝网
站
                        TestUtil.
sendRequest(url,params,secretcode,TestUtil.
HTTP_METHOD_POST,files,"UTF-8",null);
                        ……
}
```

运行结果：

首先控制台会输出：if done then input 'ok' to console!

然后会有IE弹出界面如下：

输入用户信息以后将会进入如下页面：

点击确认，然后关闭网页。在控制台中输入ok，并且回车。

此时就会发现，淘宝卖家中的一个宝贝被修改了价格

和图片，不过由于淘宝店的更新会有滞后，因此需要去我的淘宝里面看正在出售中的宝贝。

可以看到，冰激凌图片已经上传到地毯上去了，这里当然只是试验效果。■

图像的复杂度及应用

■ 文 / 温小斌

Web图片分类

构建图像搜索引擎的第一步，就是要利用爬虫程序在Internet上采集大量的图片。Web上的图片大致可分为两类：图像类和图形类。其中比较逼真的图称为图像类，这类图片一般都具有丰富的颜色信息，包括用相机拍摄的照片，用扫描仪得到的图像；图形类则具有相对简单的色彩，大部分是用电脑制作的图形、文字图片。图1中，（a）图是用电脑软件绘制的一个标志，可以看作是图形类，而（b）图则是一幅真实的风景图，属于图像类。

(a) 图形类图片　　(b) 图像类图片

图 1 两类不同的图片

网页上常见的图形类图片主要有广告图片、超链接图片、文字图片等，对于图像搜索引擎的用户来讲，这些几乎都属于“垃圾图片”，不是用户所需要的，图像搜索引擎的用户更关心的是图像类。因此，正确的把图像类和图形类图片区分开来，是构建优秀的图像搜索引擎的先决条件之一。

可以利用图片的视觉特征来区分图像类和图形类图片，这样的特征有很多，比如：

(1) 图片的尺寸。网页上尺寸太小的图片，基本都是起装饰作用的图片，不是搜索引擎的用户所需要的；

(2) 图片的高宽比。图片的高度和宽度之比值太大（或太小）的图片，基本都是广告图片；

(3) 图片所使用的不同颜色的数目。一般来讲，图形类图片中所使用的颜色总数相对较小。

然而，使用以上特征来区分图形类、图像类图片，效果不是很好，准确率不高。因此，作者在项目中引入了图像复杂度的概念，提出了基于图像复杂度的图片筛选算法，只利用一个特征，就可以很好的把图像图片和图形图片区分开来，而且，算法简单，计算量小。

图像复杂度

对于图像的复杂度，并没有一个统一的定义。不失一般性，像素值变化得越多、越频繁，则图像越复杂。因此，可以利用图像像素的变化情况来反映图像的复杂度。本文对图像复杂度的定义如下：

定义：假设一幅图像大小为M×N，则其相邻元素的个数为M×(N-1)+N×(M-1)=2MN-M-N。定义图像复杂度 α 为图像中发生变化的相邻元素的个数与图像所有相邻元素的个数之比，于是有：

$$\alpha = \frac{k}{2MN-M-N} \quad (1)$$

其中k表示图像中发生变化的相邻元素的个数。

例如一幅4×4大小的图像，其像素值为：

$$\begin{bmatrix} 100 & 50 & 50 & 100 \\ 50 & 50 & 100 & 50 \\ 100 & 50 & 50 & 100 \\ 100 & 200 & 50 & 100 \end{bmatrix}$$

根据式(1)可以计算出其图像复杂度为：

$$\alpha = \frac{16}{2\times4\times4-4-4} = \frac{16}{24} = 0.0667$$

根据上面的定义，不难看出图像复杂度 α 具有如下两点性质：

① $0 \le \alpha \le 1$；

② α 越大，表明图像越复杂。当 $\alpha = 0$ 时，图像最简单，图像中所有像素值均相同；当 $\alpha = 1$ 时，图像最复杂，图像中所有相邻像素的值均不相同。

利用图像复杂度进行图片筛选

再比较一下图1中的（a）和（b）两幅图，不难看出，图形类图片和图像类图片给人最直观的区别就是：图形类图片相对简单，而图像类图片比较复杂。用公式（1）对图1的（a）和（b）图进行计算，得到的复杂图分别为0.141和0.726。可见，利用图像的复杂度来区分图形类和图像类图片，与人类的视觉感知相吻合，是一种可行的方法。

利用图像复杂度进行图片筛选的算法很简单，只有两个步骤：

(1) 根据公式(1)计算图像的复杂度 α

Perl 6的未来（下）

上期被采访者主要谈了他的个人经历和Perl 6的开发情况，本文为续篇。

■ 文/ Kathryn Barrett　译/ 赵斯思

Terry：谈谈Perl 5的动态类型系统吧？

Damian：动态类型系统的本意是：变量只是些一般化的容器，正如你去商店买的是些什么都能装的盒子，而不是只能装毛衣、唱片或类似特定物品的盒子。然后盒子里装了东西才决定这是什么盒子，如毛衣盒、唱片盒、废料箱等。动态语言是类似的。动态语言中的值像静态语言一样有类型，但是当你把具有类型的值赋给变量时，变量会根据值来设定一些约束。这时，对这个变量，就只能进行一些对于这种类型的值有效的操作了。若将这个“盒子”中的“东西”拿出，再装进其他类型的“东西”，则这个“盒子”就有了不同的行为。这非常强大，因为这是一种延迟绑定，不用在编译器静态地做出一些决策（某些时候这些决策在后来看来并不明智）。从我们谈论的很多东西（例如面向对象）来说，如果能够延迟绑定，将决策推迟至真的获得这个对象并要对它进行操作时，那会好得多。

现今的以静态类型为主导的面向对象语言，如C++，其次是Java，你常会因为将一个动态的值赋给静态变量（它们是兼容的但是并不相同）而产生奇怪的行为和错误。动态语言中所有东西都是动态决定的。直到进行操作的时候才会去关注类型信息和检查是否被允许进行这项操作。它（动态语言）的优点在于较少的编译期不匹配造成的影响，而缺点在于直到运行期才会得到错误提示，所以它更加强调良好的测试。

Terry：所以你们在Perl 6中引入了静态类型？

Damian：是的。

Terry：为什么你们这么做？以前有反对意见吗？动机是什么？

Damian：动机是……

Terry：你仍会继续支持动态类型吧？

---

(2) 根据如下准则对图片进行分类：

if α <= Δ then

　　图形类

else

　　图像类

其中，Δ是对两者进行区分的阈值。

实验及结果

图3 网页上的图形文件

图2 网页上的图像文件

为了验证本算法的效果，作者从网易旅游频道的首页下载了一些图片来做实验，包括6幅图像类和6幅图形类图片，分别如图2和图3所示。

类别 \ 编号	1	2	3	4	5	6
图像类	0.934	0.903	0.912	0.849	0.926	0.935
图形类	0.540	0.302	0.406	0.372	0.337	0.526

表1 图2和图3中图片的复杂度比较

它们的图像复杂度对应的显示在表1中。可以看出，只要选择合适的阈值Δ，比如Δ＝0.6，就可以完全把两类图片区分开来。■

作者简介

作者简介：温小斌，硕士，现就职于上海我要印信息技术有限公司(http://www.woyaoyin.cn)，任技术总监，主要从事J2EE的开发。

Damian：是的。首先，Perl 6是一种动态类型语言，这点毫无疑问。只有值（而不是变量）才有类型信息。若你换个角度看Perl 5，它也是一种静态语言。在Perl 5中，有三种类型的对象：标量变量、数组和哈希表（对应其它语言中的关系数组或字典）。标量只能赋值给标量变量。虽然标量有字符串、数字和引用等等，但是 你只能将标量赋值给标量变量。打个比方，若有些操作是对于数组的操作，如果对标量类型进行这些操作，则会产生编译期错误。所以Perl 5有这样的静态类型，但只有三种类型可供选择。

在知道要将哪种类型的值赋值给变量而并未将此种类型的值赋值时，静态类型会在编译期给出警告，在这种情况下它有它的优势。所以，我们决定添加对静态类型支持的选项。当然，可选的静态类型支持是矛盾的，只有完全支持静态类型才行。

所以我们对Perl 6的类型系统的想法是：它默认是一个静态类型系统，你无需指定变量的类型。若你不指定变量类型，它默认是全局类型（标量、数组或哈希表）。但是你也可以指定变量类型。我想澄清的是，并不只有标量变量能存储标量，但标量变量只能存储标量。所以，当你需要静态类型时，它就在那。不需要时，它也不会烦你。

Terry：太好了！Perl 6的参数传递方式有：按位置、具名的和一种叫slurpy的方式，我们最想知道的是，什么是slurpy？

Damian：回到这个话题：最重要的是Perl 6子程序拥有参数。而Perl 5的子程序是没有参数的：所有的参数都被放进一个大数组里，你需要自己从中提取需要的。这个概念（参数）从被计算机语言实现到现在有40年了，我们终于也实现了它。

对于程序员来说，专精一到两个领域是好的，但是更重要是知识面要广，此外还必须进行实际编码。

我们都很熟悉按照参数位置传参：参数被排列好，第一个是人名，第二个是级别，第三个是序列号，按照这个顺序将它们传入。大多数语言都只提供这种传递参数的方式。这种方式的问题在于，对于只有少量参数（少于三个）的方法，它工作得很好。但是有些子程序需要八个十个或更多参数，你怎么记得起来这么多参数的顺序？ 其实你记不住，因此每次调用这个子程序的时候都得去查这些参数的顺序。所以，另一种方式是告诉子程序我有八个参数要传递，我为它们命名，若它们有了名字，那么它们的位置（顺序）就不重要了，因为我可以通过查找名字，找到人名参数、级别参数、序列号参数，如是这般。

这种具名参数传递方式在大多数语言中都没有被很好地支持，尤其是在编译器层。很多传入的是包含名字的字典或哈希表或关系数组等等，但是并没有类型检查和完整性检查，你需要自己完成这些。

Perl 6内置了对它的支持。每一个参数显然有一个名字，当调用子程序的时候，你可以将参数名写在参数值前。这样即使参数顺序不对，参数也会自动赋值。对于这种参数，有一种稍微不同的语法标识参数名，以便编译器知道重排这些参数。

当你传递参数时，你也许还想干一件事情：很多子程序有固定个数的（常常是一到两个）参数告诉它们干什么，然后有跟在后面的任意个被操作参数。一个简单的例子就是映射，将一个小函数应用到值列表中每一个值上。列表中可能只有一个值，也可能有成千上万的值。我当然不希望定义成千上万的具名参数，我希望提取第一个固定的参数，然后将接下来的所有参数生成一个数组结构。这就是slurpy参数。它吸取所有剩下的参数并形成一个容器，然后你就可以以合适的方法进行处理。

Terry：真有趣。这是Perl的一个极为函数式的特性，对吗？

Damian：是的，我们还为Perl 6引入了更好的函数式编程支持。

Terry：当前它支持映射和应用（apply），还有其他更新的吗？

Damian：全新特性里最重要的是规约（Reduce）。

Terry：规约？

Damian：没有这个操作，使用列表操作的函数编程就没终止点了，因为你始终没法得到想要的那个单一值（指映射操作无法将列表映射为单一值）。多数额外但是必要的特性并不是那么核心的内建功能，以致不能引入这种复杂的参数规格机制和函数重载机制（允许存在同名但不同参数列表的子程序）。多数函数式语言允许三个不同版本的head函数（从列表中取第一个元素），第一个版本是传入空列表，则返回空列表，第二个版本是传入单一值，则返回这个值，第三个版本是传入一个列表，返回这个列表的第一个元素（并对其它元素执行head操作）。我们将在Perl 6中也提供对这个特性的支持。

Terry：关于标记（sigil），你能说说这是什么吗？我的发音对不对，Perl 6中把它删除了吗？

Damian：我不会说来自北美的人某些发音是不是正确，因为你们把一切都弄错了。我使用的词是sigils（标记），标记是Perl中变量名开始处的噪声。这并不是Perl发明时

候的主意。如果使用过某种shell，你就知道在环境变量的开头处有一个美元标记（$）。在Perl中，标量变量在开头处有个$标记，数组有个@标记，字典（或哈希表）有个百分号标记（%）。然后可以使用同一个名称命名一个标量变量和一个数组。从一开始Perl就有这些特性，也经常出问题。这问题就是它们看起来就像是语法形态变化，若有个数组，则使用@标记，就像我们说“these things”。但是若希望谈论其中之一，我们就要说“this thing”而不能说“these things”。所以，在Perl5中，若要引用数组中的一个元素，就要改变标记。从@标记改至$标记就像从“these”改至“this”。然后使用方括号进行索引操作。从语言学来说这是个优雅的想法，它使你能敏感地分辨出不同的行为。但是实际上九成以上的编程人员无法理解这种想法。这并不是说他们笨，只是因为这并不是一种令他们觉得很自然的方法。对于标记来说，自然的想法就是，对于一个变量，以美元标记开头就是个标量，以@标记开头就是个数组，以百分号标记开头则它一定是哈希表。问题在于当改变了标记，则就改变了你使用这个变量的方式，这很让人困惑，尤其是说英语的人，因为我们并不常常这么做（意指英语里这样的用法很少）。

当你改变你使用单词的方式时，其他语言有更多的语法形态变化，而英语就不是这样。我们发现很多用户无法理解对于同一个变量的标记的改变。所以，我们改变了Perl 6中对于标记的处理。在Perl 6中，变量名开始处的标记总是相同的。若它是一个数组，则@标记不会被改变，无论你如何使用它——无论你在其中查找，获取子数组等等。这是Perl 6的一个非常重要的改变。它将挑战有经验的Perl 5程序员。但是如果你用惯了这个特性，就再也不想回到从前了。它比原来好太多了。

Terry：我前面问过什么书你觉得对于有理想的计算机科学专业的学生是最好的。更一般地，你能给年轻的有理想的程序员一些一般的建议吗？我想前面你已经谈及这个部分。

Damian：我想我的建议（如果保持和前面的论调一致的话）是：对于程序员来说，专精一到两个领域是好的，但是更重要是（知识面）要广，至少要大概了解各种各样的做事的方法，多样的编程风格，多种编程语言，和各种不同的算法，这是你的知识储备。在某些特定的情况下，时间很紧张而你无法在短时间内想出妙招，这时你就能依靠它。这时你可能就会说，如果我们使用Haskell就能解决这个问题，我们能引入Haskell吗？并不是说一定要用Haskell解决这个问题，但是你能想到这个解决方法吗？我想这才是真正重要的。要成为优秀的程序员，你必须先成为一个见多识广的程序员。

我想说的另一件事情是，要成为优秀的程序员，你必须进行实际编码。这可能是不会发生的事情（意指可能很少编码）。我们经历过学生生涯，我们入学并学习所有东西，我们不断地练习和考试等等。然后毕业了，开始参加会议、做各种设计及一些别的工作而不再编码。如果获得升迁，实际上可能就失去了编码的机会。我认为这就是问题。如果想成为真正优秀的网球运动员，你会每天出去练习。如果想成为优秀的武术家，你会每天呆在武馆里。如果想成为优秀的程序员，你就要每天编码，即使你使用自己的时间做这件事情。 当你起床你就是个程序员，无论是晚上11点起床还是凌晨3点起床，至少有部分时间用来编码。若逐渐荒废，则意味着离程序员也越来越远。

Terry：太好了。非常感谢。这就是Damian Conway，非常感谢。

Damian：谢谢。■

名人故事　唐宗汉与Perl 6

唐宗汉，1981年生于台湾，14岁辍学，说话语速极快。他自从小学二年级写出第一个数学软件以后，就迷恋上了电脑，写程序和玩电脑成了他生活的全部。学习Perl只是由于正值互联网的兴起，开始他觉得写Perl很孤独，但是经过长时间的与各种朋友的交流才发现，原来很多的软件都是用Perl写，很多大公司都在使用Perl，例如：摩根士坦利整个公司都在使用Perl，他们在2003年赞助了Perl基金会大约一半的费用；众多华尔街的金融机构都在使用Perl来做金融数据的处理。生物工程科学也都在很大程度上依赖于Perl。

2000年，由Larry wall在每年一次的开放源代码大会（oscon2000）上提出了Perl 6的开发计划。它的一个特性就是把当时最流行大家最推崇的其他语言好的功能集合起来，统一融合到自己的语言中来。在Perl中这叫多重典范。Larry wall喜欢把这种情况比作英文，英文就是吸收了很多其他语言而组合而成的语言。每个人在学习英文的时候都会带一些当地的方言，带一些当地的讲话习惯，但是大家也都能懂。Perl还有另外一个特点是语境相关(context)。但是在实现Perl 6的时候遇到了问题，为了更方便地把新功能加入Perl，需要实现自举，以前的Perl是用C写的，这回需要用Perl写。Damian编写了的Perl6::Rules模块，Perl 5上实现了Rules的雏型，但还不是完全能解决这个问题。

这个时候，大概是在2005年2月唐宗汉开始了一件很疯狂的事情：对于过去的Perl 6开发通通都不管，从新编写一个解释器pugs直接运行Perl 6。整个项目以一种很恐怖的速度开发，第7天的时候就可以跑大部分的运算程序，到了第20天就有了模块，到了第22天就有上千个测试。现在，我们无论怎样评价pugs项目，它在事实上已经成为Perl 6的转机。pugs这个项目是用haskell，一种纯正的函数式程序设计语言来写的。懂haskell的人不是很多，唐宗汉也不过只学了两个月，现在，他正在号召Perl社区的人学习haskell。

■ 责任编辑：赵健平（zhaojp@csdn.net)

Perl在生物研究中的应用

实验生物学家通常要不断地面对大量的文件，这些文件体积硕大、格式不相兼容。本文叙述了在这一领域中使用Perl的经验。

■ 文 / Amir Karger，Eitan Rubin　译/韩锴

问题：脚本的源头

实验生物学家通常要不断地面对大量的文件，这些文件体积硕大、格式不相互兼容。生物学家们需要对这些文件进行过滤、重格式化、合并、分割（定义3）的操作。不会编写Perl程序的生物学家们，大多都是如此，最终都是手工地编辑这些文件。如果他们在一周后遇到了问题，**就必须重新做相同的事情**——或者只能放弃。

我的工作就是帮助这些生物学家们使用计算机。我可以给他们写一个考究的、一次性的脚本，是不是真的这样呢？在公布答案之前，让我给你讲一讲 Neeraj的故事。一天，Neeraj一位典型的NPB（不会编程的生物学家），他来到我的办公室，对我说“我有12000个（碱基或核苷酸）序列，需 要‘make primers for’。”“要什么？”我问。（我可从没有长时间地从事生物学研究）。幸运的是，他很快就解释清楚了他的目的——他要做的无非就是从一个文件中取得每个DNA的第201-400个字符。这对于已经浸染Perl多时的你来说，闭着眼睛就可以完成了（如果你能摸到键盘的话）：

```
perl -ne 'print substr($_, 200, 200), "\n"'
sequences.in >
primers.out
```

三下五除二，我把输出文件交给了Neeraj，他高兴地离开了，继续组建他的克隆军队，并占领世界（或者他是去改变大米的基因来解救世界的饥饿，我不是清楚）。不幸的是，事情还没有结束。第二天，Neeraj又回来了，这次他又想从后面开始截取相同数量的字符 (substr($_, -400, 200))。 由于他在从事一项尖端的研究工作，因此在他下个月完成实验的时候，完全可能有新的需求。在这个团队里，只有几个人为数百甚至数千名生物学家提供支持，编写考究的（甚至是单行的）脚本，并不具有伸缩性。其他常见的办法，比如教会生物学家们如何使用Perl，或者开发一个图形化的工作流管理器，似乎不能完全解决数据操作的问题，尤其是对那些偶尔才会分割数据的用户。我们需要一些工具，帮助Neeraj和其他任何一位NPB可以自己分割手头的数据，而不必依靠一个程序员，而且你还要教授他生物学。这样生物学家们就能最好地应用数据和算法来解决问题。这样的工具对于非程序员来说应易于学习和记忆，应该是TMTOWTDI（There is more than one way to do it）的，应该允许使用者不断地对数据进行实验，直到他们找到最适合的处理办法。尽管有我们这些程序员在，这样的工具仍然需要能够应付快速变化，以满足生物学家们不断提出的新挑战。当我把Neeraj的故事讲给团队里的其他人后，他们说自己和这个问题已经斗争好几年了。在一次头脑风暴会议里，我的老板Eitan Rubin说：“我们为什么不为生物学家们提供一本写满可以用来数据分割的脚本的魔法书呢？”“啊，一种Script Tome？”于是，Scriptome诞生了。

利用原子的能量

The Scriptome是一本手册，用于生物学领域的data-munging(数据分割)。它与UNIX的风格非常吻合——保持每个工具小巧，并只完成一件简单的事。不过我们摒弃了UNIX的管道，而是使用中间过渡文件，这是为了避免出现错误。为了让NPB能够利用这本手册，我们使用了很多小技巧。我们选择大家熟知的Web浏览器当作GUI，并利用超链接为整套工具开发了一个模块化的、清 晰的目录。这就是说，即使工具的数量从几十增长到几百个，也不需要用户记住命令的名字。另一个小技巧是语法高亮。我们灰化了大部分Perl，表明它们只是可选的。对于参数（比如文件名、最大值等等），我们将它设置为红色，以提醒使用者的注意。最后我们花了很多努力避开计算机科学的术语。我们使用的语言都是生物学家们所熟悉的。比如，把工具称为“原子（atom）”，而不是“片段（snippet）”。

每一个Scriptome工具都包含了一个（在彩色框里的）

简单的Perl程序、几句话的文档（如果再多一些的话，就没有人再读它了）以及输入和输出的范例。要想是使用一个工具，你需要：

- 选择工具的类型。比如需要挑选文件中的某些行或者列，那可能要选择“Choose”。
- 浏览一个层次化的目录。
- 从一个彩色框里面剪切一些代码，并粘贴到Unix、Mac OS X或者Windows的命令行中。
- 使用方向键或者文本编辑器，将红色的文本改为期望值。
- 点击Enter。
- 就这些！

分解Scriptome

我前面向你们讲的Neeraj的故事并不是完全准确。实际上他要同时打印每个序列的开始和结尾的子字符串。另外，他的输入的是FASTA格式的，也就是每一个序列都有一个ID行（比如>A2352334）放在DNA字母的前面。我们没有现成的工具能够解析FASTA格式并获取两个不同的字符串；如果写出所有可能的组合，恐怕花费的时间比实现Perl 6还要长。因此，我们又一次遵循UNIX的规矩，让生物学家们针对特定的问题，自己组合工具，形成相应的解决方案。在这个例子中，解决方案将涉及到FAST-to-table的转换器，一个从序列中提取列的工具，以及两个substring工具。

我们要求生物学家将问题进行划分为更小的问题——每个问题都可以通过一些工具加以解决——然后使用正确的参数、按照正确的顺序将它们组装起来。这听上去与编程惊人地相似，不是吗？虽然你可能不会考虑很多，可编程的一些基本概念是新鲜而困难的。幸运的是，生物学家们已经知道如何分解问题、循环、分支和调试；当然他们不把这些称为“编程”，而是“流程（protocol）设计”。一个流程可能包括下面几行：

1. 在DNA中加入1 ml的XXX生物酶。
2. 在试管中放置一小时，保持90度。
3. 如果混合物澄清，转到步骤5。
4. 重复三次步骤2-3。
5. 将液体非常小心地倒入消毒瓶。

在这里借用术语“流程（protocol）”来描述一个有序的、参数化的Scriptome工具集，它能够解决更大的问题。（真正的东西其实是“脚本”，但是不要告诉我们的客户，否则他们会觉得自己是在学习编程。）我们预先在Website上写好了一些流程。注意，每个工具都是命令行，而把它们放在一起是个不折不扣的可执行的Shell脚本。对于NPB来说，Scriptome不仅仅是一个高级的、语法自由的、非玩具的语言。因为它直接在Website上暴露了Perl——赋予“Open source”一种新的内涵——一些具有好奇心的生物学家会逐渐尝试阅读短小的、简单的Perl代码。

智慧的设计：细节

如果你读到这里了，可能已经发现，Scriptome在本质上并不是一个程序设计项目。设计、接口、文档和范例都与编程本身同等重要。编程是很容易的。然而我要谈一谈整体而言，项目总体是如何使用Perl的。

为什么选择Perl?

很多人问我为什么没有用Python、Ruby来写Scriptome，或者干脆用UNIX sh解决所有问题。好的，除了一些显而易见的理由以外，它还非常适合快速工具开发，可以移植到了很多平台上，它几乎已经被安装到每一台Unix和Mac OS X机器上了。而且，Bioperl模块给我提供了大量可重用的工具。最后，Perl是整个Scriptome开发团队（我）最喜欢的语言。

哪种类型的Perl?

只需要几百个字符，你就能用Perl编写出令人印象深刻的工具来。另一方面，我们希望代码能够具有可读性，尤其是我们希望新手能够通过它来学习，那么就不能使用过多的Golf快捷表达方式。例如，下面这段Juho Snellman编写的脚本使用了Perl Golf快捷表达方式，它能够查找N因式分解后的最后一个非零数字：

```
#!perl -l $_*=$`%9e9,??for+1=~?0*$?..pop;print$`%10
```

这样的代码对于新人来说太难了。

构建Scriptome

尽管我们试图保持工具尽量通用和简单，但是仍然需要几十个这样的工具。另外，数据的格式和生物学家们的兴趣也不断会变化。因此我们必须保证创建新工具的过程快速与自动化。

我用POD编写Web页面，这样我可以使用Vim而不是一个Web-page编辑器。Makefile会运行pod2htm创建一个相当棒的web页面，它包含了目录表格。一个Perl过滤器会添加一个导航条和一些改善界面的JavaScript，并把参数设置为红色。我会让步，转而使用一个模板化的系统，数据库后端，带有自动化测试就更棒了。就现在而言，”保持简单”意味着我可以在一个小时以内创建、测试、添加文档并最终发布一个新的工具。

Perl文化

项目里面已经有了很多Perl代码，但是我还试图在里

面增加一些Perl的态度。在Scriptome项目的过程中，每当我们意识到现在只需要几行代码就能够帮助生物学家完成任务，而不必花费6个月或者一年去打造一个完美的解决方案时，就是产生”Aha!”感觉的时刻。尽管很多计算生物学家更关注于编写一个复杂度为O(N)的算法来分析序列或者研究基因表达式，我们并不为编写胶水程序而尴尬；我们解决那些看上去没有吸引力的问题。毕竟，如果data munging是Neeraj研究中的一步，那么没有这些工具，他也无法发表他的论文。最后，我们正在倾听来自客户的略带挑战味道的意见，因为只有他们才知道哪些简单的事情可以简化，哪些复杂的问题是可能被解决的。

填补空白：Scriptome和其他的方案

与生物学家们讨论data munging时，最大的顾虑在于他们甚至没有认识到问题的存在，或者他们认为已经把问题解决了。生物学家——一群擅长用吸管不断重复的人——不会意识到计算机能够为他们节省大量的时间。很多程序员只要读一读《Learning Perl》这本书就足以胜任了。我就是这样做的。但是实验生物学家花费在收集数据上的时间远远多于分析数据的时间，因此他们无法承受程序员去占用他们的时间。他们不必如此。每一个生物学家都需要多继承、getprotobyname()和正则表达式吗？其实有很多繁杂的问题需要简单的、灵活的工具去处理，但这些问题又过于简单，不需要大动干戈地去编程。上一节三个小时的Perl入门课如何？这门课上，至少要讲授变量、数组、哈希表、正则表达式和控制流——当然，还有语法。“等等，能不能再讲一遍@{$a[$a]}和@a{$a[$a]}之间有什么不同？”“哦，看看时间吧。”Damian Conway曾经写过一篇文章，名为《引导程序设计语言的七宗罪》，里面谈到了奇怪的语法通常会令新手分心，无法专注到基本的语言概念上。三个小时里你能教多少内容？没有实践，一个月后你的学生还能记住多少？

另一种方法是构造一个图形化的程序，让生物学家们完成所有事情。这样，管道就变成了拖拽图标和连接器的操作了。然而，一个完善的图形化环境需要花费很大努力去构建。除了这一点以外，一个全功能的、图形化的程序必然会非常复杂，提高了初学者的门槛。

在构建Scriptome的过程中，我们有意地窄化了问题领域，为那些偶尔使用的用户最大化了可学习性和可记忆性。尽管讲授编程技能和创建图形化的工具在有些时候有效的解决方案，但我相信，Scriptome仍然填补了data munging领域的一项空白。

“创造”非易事

在解决工具使用与程序设计之间，我们取得了很大的进步。早期的审阅者们留下的多是正面的评价，至少是建设性的。第一位用户Suzy起初抱有强烈的怀疑，说她可能要学习Perl了，因为我们提供给她的任何工具都不够灵活。我鼓励她学习Perl的工程中同时使用Scriptome。最终的结局是，她自称为“Scriptome专家”，能够从工具中提取出代码，然后创建一个16步的流程，这是真正的生物学研究工作。我们优先级最高的事情就是构建一个用户基础，从中获得关于Website及大约50个工具的可学习性、可记忆性和有效性的反馈。

这个项目还需要更多的工具、新的流程以及可能的新的界面。你现在就可以为新工具写一些代码。我们希望听到各种不同的想法和主意。

下面有一个小小的挑战。我真的遇到过一个人，他手工地为768个文件重命名。那时我没有给他写Perl脚本。你能为NPB写一个通用的重命名脚本吗？（提示：“告诉用户去学习正则表达式”是行不通的）胜利者将会得到Scriptome的一个纪念奖牌(<bgcolor="gold">)。

谈到新的界面，我们听到的一种普遍担忧是NPB不能或者不愿意和命令行打交道，而Scriptome要用到一些命令（cd, more, dir/ls）。于是我们的用户说，这是个问题。我们正在探索一些不同的方式去封装Scriptome工具，比如：

- Firefox插件。在工具栏中提供命令行输入，在浏览器中显示结果。（当前正由Rob Miller和他在MIT的团队负责开发）
- Excel VBA。在某一栏中输入命令，然后创建一个Shell脚本。
- 使用Pise或者GenePattern封装命令行。

我们会尝试上面这几种途径，因为它们允许用户在希望的时候继续使用命令行接口。

谈到未来么...恩，谁说只有生物学家才需要分割数据？想一想，化学家和天文学家同样要做这些工作。我要把眼光放得更远一些。对于一个需要分割报告的业务经理来说，Scriptome能不能派上用场？用“Apache Scriptome”管理你的网站访问记录如何？用“iTunes Scriptome”管理你的音乐如何？对于那些要操作数据的人来说，Scriptome可以给他以力量。

原文链接：http://www.perl.com/pub/a/2005/10/20/scriptome.html ■

作者简介

Amir Karger将计算机技术与生物科学两大尖端科技有机地结合在一起，目前为生物学家的研究工作提供生物信息学方面的支持。

■ 责任编辑：赵健平（zhaojp@csdn.net）

放大愉悦——《征途》策划新理念

■ 文 / 萨菲罗斯

就网络游戏的玩家群体而言，追求复杂的系统与深度内涵的CORE-PLAYER，与崇尚简单、轻松的LIGHT-PLAYER是最常见的两类，随着WEB2.0时代的到来，休闲网游、网络社区、网页游戏的逐渐兴起，越来越多的LIGHT-PLAYER加入到网游玩家的行列中来。

虽然RTS、ACT、FPS等新类型网络游戏正在不断涌现，但传统的RPG网络游戏仍然是目前市场上研发与运营的主流，面对三类玩家的不同需求，RPG网游的策划不得不“适时而动”。

史玉柱自2004年进入网游市场，四年创业使巨人网络成为立身国内网游市场前列的开发商与运营商，其主持策划的《征途》以黑马之姿最终成长为一款平均在线过百万的市场主流游戏。本文试图通过从游戏策划的角度分析《征途》来探讨史玉柱的游戏策划理念。

定位愉悦

一款游戏畅销风靡的原因有许多，其中“游戏性”是被讨论最多的话题。

“游戏性”是评价游戏的一个关键参数，主要体现在三个方面：趣味性（愉悦来源）、严密性（数据平衡）、耐玩性（反复尝试的冲动）。对于网络游戏来说，趣味性因网络世界的多样性而来源更多更广泛，平衡性因玩家之间互动的普遍化而更显关键，耐玩性因网游内容的不断添加比重下降。

显然，就网络游戏“游戏性”而言，数据平衡是一个必须满足的物理化的硬指标，而愉悦来源成为衡量游戏综合实力的主要因素。成功的游戏策划，需要衡量三类网游玩家的不同口味，设置不同比重的愉悦要素，既主题突出，又兼收并蓄。

具体而言，网络游戏的愉悦来源主要有：

1 取得进步：通过提升升级、收集金币等不断增强实力

2 PK：与其他玩家作战并取得优势地位

3 聊天交友：在游戏中认识并与其他玩家交朋友

4 团队协作：与其他玩家组队、合作

5 掌控游戏机制：通过游戏分析和理解系统内在的数值系统

6 探索：探索游戏世界，发现稀有的区域、任务或物品

7 角色扮演：RPG网游独有的愉悦要素，以NPC的

征途游戏场景：别墅

视角融入游戏世界

网络游戏早期，CORE-PLAYER是主要的群体，对于这一类型的玩家来说，取得进步和PK可谓唯一的愉悦来源，交友、组队、探索、掌控系统均处于辅助与次要的地位，都是为掌握快速升级的“钥匙”并最终获得PK胜利

征途同城约会资料片

而服务的。CORE-PLAYER拥有充足的时间和旺盛的精力，能够承受重复性很强的练级和操作，甚至乐在其中。《传奇》的火爆就在于其游戏架构很好地满足了这一类型玩家的愉悦需求，其呈现为单机RPG游戏的常态——以等级为贯穿始终的线式结构。一定范围的区域、怪物、技能、装备结合为一个个模块，模块之间通过等级串联，玩家之间通过PK或组队挑战大型MOB串联。由于玩家与模块的一一对应关系令高等级玩家很少回顾低等级模块，《传奇》后期升级的海量经验也就不足为奇了。

随着平均在线时间2小时以内的LIGHT-PLAYER进入网游的视野，网络游戏的在线人数开始“爆炸性”增长，几十万在线的游戏频频出现。为了满足两类玩家的不同需求，聊天交友、团队协作、探索地图等愉悦来源开始逐渐摆脱从属地位。

对于LIGHT-PLAYER而言，游戏策划则主要呈现为聊天系统的不断进化和休闲功能的不断添加，这也是《征途》的主要策划方向之一，容下文详述。

而对于CORE-PLAYER而言，网游的策划方向开始出现分化。一是以《大话西游》为代表的转生+宠物系统的引入，重点提供取得进步这一愉悦来源，游戏架构呈现为传统的线式结构；二是以《魔兽世界》为代表的团队副本系统的引入，重点提供团队协作这一愉悦来源，游戏架构呈现为等级+副本的层式结构。两种架构都可以最终服务于享受PK的愉悦。

基本上，RPG网游对以上要素都有所借鉴，区别不过是借鉴力度的不同而已。

《征途》被史玉柱定位为“休闲+PK”的RPG网游，1到200级的等级设定表明其游戏架构仍然是传统的“线式”。不同之处则在于，《征途》面向LIGHT-PLAYER的需要，添加了多种休闲功能，强化了PK功能，并以聊天交友对其他愉悦要素进行全面“渗透”。

简化操作

《征途》吸引LIGHT-PLAYER的第一步，是简化操作

没有人生来就是喜欢打字的，玩家在进行游戏操作时，是得不到任何乐趣的，给他们乐趣的，是操作的结果。所以，操作在游戏过程中，是一种额外付出的行为，甚至可以说，是玩家为了获得愉悦所付出的“代价”。一般说来，游戏操作越便捷，越令人感觉可以忽略不计，也就越成功。

我们可以用这个公式来说明：游戏性＝游戏整体-操作

上式所指的操作，主要是游戏过程中与主要愉悦来源无关的大量重复性操作，比如装备修理、买卖药品、往返安全区与MOB区的行路操作以及练级时的大量重复性攻击操作等等。显然，LIGHT-PLAYER很难如CORE-PLAYER一样忍受这些繁琐、复杂的重复性操作甚至从中收获愉悦，想要吸引大量的LIGHT-PLAYER，必须尽可能地简化操作。

这里要强调的是，简化操作绝不意味着无视操作时间，比如《魔兽世界》允许辅助操作的插件存在，但绝对不会容忍瞬移类型的游戏外挂。很多游戏都通过增设传送/复活点、一键修理、一键换装等方式缩短操作时间，但原地复活、替身娃娃等无视时间耗费的设计则只会以增值服务的方式出现，就是由于这个原因。

LIGHT-PLAYER很少能够从游戏角色在地图上跑来跑去的过程中收获愉悦，尽管随着网络游戏地图疆域的日趋扩大这种时间耗费具有必然性，但LIGHT-PLAYER需要从移动、攻击等重复性的时间耗费中找到其他的乐趣。

《征途》是以“内挂”这一游戏设计来解决这一问题的。首先是添加了自动移动的功能，并通过大地图点击移动、坐标输入移动、NPC搜索移动、任务目标超文本链接移动等多种方式来便捷操作；其次是添加了自动攻击的功能，并通过自动补充HP/MP、自动攻击附近MOB、自动拾取物品等功能实现攻击操作的大幅度简化；此外，由于添加了挂机功能，以增值服务方式实现了离线获取经验值。

在“内挂”这种游戏设计中，练级与PK/团队协作被人为划分为两个截然不同的阶段。在练级过程中，玩家更多地是以一个游戏人物的管理者而非操作者的身份出现，只有在玩家之间的PK或者团队协作过程中，玩家才需要关注游戏人物的操作。

对于LIGHT-PLAYER而言，“内挂”实现了练级的后台操作，极大地降低了玩家的精力耗费。而内挂的“成本”，则是对战普通MOB的愉悦值大幅降低，尽管《征途》仍然是传统RPG网游的线式架构，但取得进步这一愉悦来源的地位已经大大降低，相应的PK则上升为第一位。《征途》之所以定位为“休闲+PK”，不仅是迎合LIGHT-PLAYER的主动出击，也是作为核心设计的“内挂”系统的客观使然。

“休闲+PK”练级

游戏的线式架构，意味着等级的增长处于整个游戏的“主线”地位，玩家在游戏中仍然需要不断积累经验值、提升等级，然而“内挂”设计在降低练级精力耗费的同时，又不得不承担练级愉悦值降低的代价。

《征途》的应对办法，是在FARM MOB之外，休闲游戏和玩家PK同样可以获得经验值。

传统RPG网络游戏的“玩点”多脱胎于单机版RPG游戏，主要包括等级、人物、装备、宠物等要素，而对其他类型的单机游戏，如TAB（桌面游戏）、AVG（冒险游戏）、ACT（动作游戏）、RTS（即时战略游戏）则借鉴不多。《征途》休闲练级的一大来源，就是大量引入其中游戏类型的要素，以小游戏的方式获取经验值，比如运镖、斗地主、跳舞、自行车比赛、采集、智力答题，等等。

网络游戏区别于单机游戏的最大特点就是普遍的人际交互，聊天交友也是玩家获得愉悦感的主要途径，对于LIGHT-PLAYER尤其如此。有鉴于此，《征途》也将聊天交友系统融入获取经验值的过程，添加了很多交友练级的设计，比如情侣沙滩、同城频道、好友任务等等。

《征途》设计了多样化的群体PK方式，在这些PK战中，玩家都可以通过击杀敌对玩家、敌对NPC或完成团战任务来获取大量经验值。目前，《征途》已经推出家族NPC争夺战、帮会夺城战争、国家骚扰战争、国家正式战争、国家奇袭战、中立区夺城战、战场竞技、大海战等多种群体PK方式。

巩固社区

除了休闲游戏之外，网游玩家在聊天交友中收获的愉悦感也非常大，对于他们来说，在游戏中认识朋友，就已经是在“游戏”。

正因为如此，社区化也成为《征途》策划理念的核心，通过强化聊天交友功能，使之不仅是游戏的一种功能、一个系统，还要将聊天交友融入游戏的其他愉悦要素，使聊天交友成为游戏的一种“平台式”的存在。

就网络游戏玩家个体而言，高等级玩家与低等级游戏模块的联系微乎其微，当一款网络游戏不再更新的时候，也就意味着高等级玩家的离开。为了延长游戏的生存寿命，策划人员引入了转生系统和宠物系统，转生意味着一次小开“金手指”新人之旅，宠物则意味着另一种含义的练级。而无论是转生还是宠物，都是在原有游戏架构的基础上所做的改进，可一而不可再。

《征途》将交友的愉悦与取得进步的愉悦相融合的一个体现，就是“师徒”的设计。在《征途》的荣誉系统中，高等级玩家可以与50级以下的新玩家结为“师徒”，并通过带徒弟练级获得荣誉值，荣誉值不仅可以消去罪恶值、交换道具，还能够换得游戏币。这一设计通过新老玩家的交友功能，以代练的方式，沟通高级玩家和低级模块，达到延长游戏寿命的目的。

大量LIGHT-PLAYER涌入网络游戏，决定了游戏策划必须向巩固游戏社区的方向倾斜。提供更多样化的玩家交互手段，支持更多种类的玩家结社方式，这些都是在策划层面巩固线上社区的尝试。

不过，玩家之间聊天交友不仅意味着在游戏中收获愉悦，这种关系还将延伸至线下，由游戏中的好友而至现实中的朋友，线下关系的紧密对线上群体的影响更加明显，见过面的网友、知根知底的网友、纯粹的线上网友，其亲密度绝对不能相提并论。

因此，从网游社区的理念出发，将聊天交友系统的作为立足点，由方便玩家交友转变为促使玩家交友，显然更具有主动性，更有利于玩家社群的巩固。《征途》的第三部资料片“同城约会”，就提出了“线上交友、线下见面”的理念，通过开设省聊/市聊频道、专门的好友空间、情侣沙滩等，使同地区的玩家在游戏中结识，在现实中交友。

史玉柱自称将自己十几年游戏生涯的感受和想法都注入了《征途》，并认为《征途》的成功缘于自己对玩家心理的准确把握。上述分析也表明，《征途》“休闲+PK”的定位，确实契合了CORE-PLAYER对PK的追求，也满足了LIGHT-PLAYER对休闲交友的需要。■

■ 责任编辑：赵健平（zhaojp@csdn.net）

主持人：张银奎
《软件调试》一书作者，英特尔亚太研发中心高级软件工程师。从事软件开发和研究十余年，对IA-32架构、操作系统内核、虚拟技术，尤其对软件调试有较深入研究。翻译（合译）作品包括《数据挖掘原理》、《机器学习》、《人工智能：复杂问题求解的结构和策略》等。

步步为营
——如何调试操作系统加载阶段的故障

上一期我们介绍了系统固件（BIOS）寻找不同类型的引导设备的方法，描述了固件向引导设备移交执行权的过程。对于从硬盘引导，首先接受控制权的是位于硬盘的0面0道0扇区中的主引导记录（Main Boot Record），简称MBR。MBR一共有512个字节，起始处为长度不超过446字节的代码，然后是64个字节长的分区表，最后两个字节固定是0x55和oxAA。MBR中的代码会在分区表中寻找活动的分区，找到后，它会使用INT 13h将活动分区的引导扇区（Boot Sector）加载到内存中，加载成功后，将执行权移交过去。按照惯例，引导扇区也应该被加载到0x7C00这个内存位置，所以MBR代码通常会先把自己复制到0x600开始的512个字节，以便给引导扇区腾出位置。也正是因为这个原因，当使用虚拟机或者ITP调试时，如果在0x7C00处设置断点，那么这个断点通常会命中两次。引导扇区的内容是和操作系统相关的，在安装操作系统时，操作系统的安装程序会设置好引导扇区的内容。引导的职责通常是加载操作系统的加载程序（OS Loader）。OS Loader得到控制权后，再进一步加载操作系统的内核和其它程序。本期我们就以Windows Vista操作系统为例谈一谈OS Loader的工作过程以及如何调试这一阶段的问题。

切换工作模式

我们知道，对于x86 CPU来说，不管它是否支持32位或64位，在它复位后都是处于16位的实地址模式。在BIOS阶段，CPU可能被切换到保护模式，但是在BIOS把控制权移交给主引导记录前，它必须将CPU恢复回实模式，这是一直保持下来的传统。对于使用EFI固件的系统，固件可以在保护模式下把控制权移交给操作系统的加载程序。但本文仍旧讨论传统的方式。

因为实模式下的每个段最大只有64K，而且只能直接访问1MB的内存，这个空间是无法容纳今天的主流操作系统的核心文件的，所以OS Loader首先要做的一件事就是把CPU切换到可以访问更大空间的保护模式。

在切换到保护模式前，应该先建立好全局描述符表（GDT）和中断描述符表（IDT）。通常在OS Loader阶段不会开启CPU的分页机制（Paging），而且描述符表中的每个段的基地址通常都设置为0，界限设置为0xFFFFFFFF，这样便可以在程序中自由访问4GB的地址空间，而且线性地址的值就等于物理地址的值，这里使用内存空间的方法就是所谓的平坦模式（Flat Model）。以Windows Vista操作系统为例，它的引导管理器程序BootMgr.EXE内部既有16位代码又有32位代码，16位代码先执行，在验证文件的完好后，会切换到保护模式，并把内嵌的32位程序映射到0x400000开始的内存区，然后把控制权移交给32位代码的起始函数BmMain。此时观察CR0寄存器，可以看到代表保护模式的位0已经为1。

```
kd> r cr0
cr0=00000013
```

但是代表分页机制的位31为0，说明没有启用分页。观察代码段和数据段的段描述符：

```
kd> dg cs
                                   P Si Gr Pr Lo
```

```
Sel    Base     Limit     Type    l ze an es ng
Flags
---- -------- -------- ---------- - -- -- -- --
--------
0020 00000000 ffffffff Code RE Ac 0 Bg Pg P  Nl
00000c9b
kd> dg ds
                                  P Si Gr Pr Lo
Sel    Base     Limit     Type    l ze an es ng
Flags
---- -------- -------- ---------- - -- -- -- --
--------
0030 00000000 ffffffff Data RW Ac 0 Bg Pg P  Nl
00000c93
```

可见，它们的基地址都是0，边界都是0xFFFFFFFF，这正是平坦模式的典型特征。分别使用dd命令和!dd（观察物理地址）观察同一个地址值：

```
kd> dd idtr l4
0001f080  00500390 00008f00 002073b0 00448e00
kd> !dd idtr l4
#   1f080 00500390 00008f00 002073b0 00448e00
```

显示的内容是一样的，这说明线性地址与它所对应的物理地址的值是相等的。

休眠（Hibernation）支持

在执行BlImgQueryCodeIntegrityBootOptions函数和BmFwVerifySelfIntegrity函数对自身的完整性做进一步检查后，BootMgr会调用BmResumeFromHibernate检查是否需要从休眠（Hibernation）中恢复，如果需要，那么它会加载WinResume.exe，并把控制权移交给它。

显示启动菜单

BootMgr会从系统的引导配置数据（Boot Configuration Data，简称BCD）中读取启动设置信息，如果有多个启动选项，那么它会显示出启动菜单。清单1中的栈回溯显示的便是BootMgr在显示启动菜单后等待用户选择时的状态。

```
kd> kn
 # ChildEBP RetAddr
00 00061e34 00432655 bootmgr!DbgBreakPoint
01 00061e44 00431c24 bootmgr!BlXmlConsole::
getInput+0xe
02 00061e90 00402e8f bootmgr!OsxmlBrowser::
browse+0xe0
03 00061e98 00402b5e bootmgr!BmDisplayGetBootMenu
Status+0x13
04 00061f10 004017ce bootmgr!BmDisplayBootMenu+0x
174
05 00061f6c 00401278 bootmgr!BmpGetSelectedBootEn
try+0xf8
06 00061ff0 00020a9a bootmgr!BmMain+0x278
WARNING: Frame IP not in any known module.
Following frames may be wrong.
07 00000000 f000ff53 0x20a9a
08 00000000 00000000 0xf000ff53
```

清单1 等待用户选择启动项

栈帧6中的BmMain便是BootMgr的32位代码的入口函数，栈帧4中的BmDisplayBootMenu是显示启动菜单的函数，栈帧7和8是在实模式中执行时的痕迹。

执行用户选择的启动项

当用户选择一个启动选项后，BootMgr会调用函数来准备引导对应的操作系统。如果系统上有Windows XP或者更老的Windows，而且用户选择了这些项，那么BootMgr会加载NTLDR来启动它们。如果用户选择的是Windows Vista的启动项，那么BootMgr会寻找和加载WinLoad.exe，如果没有找到或者在检查文件的完整性时发现问题，那么BootMgr会显示出图1所示的错误界面。

在成功加载WinLoad.exe后，BootMgr会为其做一系列其它准备，包括启用新的GDT和IDT，然后调用平台相关的控制权移交函数把执行权移交给WinLoad。在x86平台中，完成这一任务的是Archx86TransferTo32BitApplicationAsm函数。至此，BootMgr完成使命，WinLoad开始工作。

加载系统核心文件

WinLoad的主要任务是把操作系统内核加载到内存，并为它做好“登基”的准备。它首先要做的一件事就是进一步改善运行环境，启用CPU的分页机制。然后初始化自己的支持库，如果启用了引导调试支持（稍后介绍），那么它会初始化调试引擎。

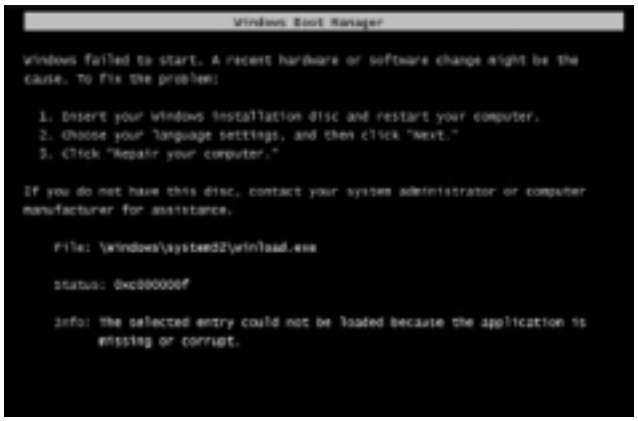


图1 加载winload.exe失败时的错误提示

接下来WinLoad会读取启动参数，决定是否显示高级启动菜单，高级菜单中含有以安全模式启动等选项，也叫Windows Error Recovery菜单。如果用户按了F8或者上次没有正常关机，那么WinLoad便会显示高级启动菜单。

接下来要做的一个重要工作是读取和加载注册表的System Hive，因为其中包含了更多的系统运行参数，负责这项工作的是OslpLoadSystemHive函数。

做好以上工作后，WinLoad开始它的核心任务，那

就是加载操作系统的内核文件和引导类型的设备驱动程序。它首先加载的是NTOSKRNL.EXE，这个文件包含了Windows操作系统的内核和执行体。此时真正的磁盘和文件系统驱动程序还没有加载进来，所以WinLoad是使用它自己的文件访问函数来读取文件的。例如，FileIoOpen函数便是用来打开一个文件的，

如果FileIoOpen打开文件失败，那么调用它的BlpFileOpen函数会返回错误码0C000000Dh，否则返回0代表成功。

接下来加载的是硬件抽象层模块HAL.DLL，支持调试的KDCOM.DLL和它们的依赖模块。使用Depends工具可以观察一个PE模块所依赖的其它模块，例如，图2显示出了内核文件NTOSKRNL.EXE所依赖的其它模块。

其中，PSHED.DLL用于支持WHEA（Windows Hardware Error Architecture）（《软件调试》第17章有详细介绍），HAL.DLL是硬件抽象层模块，BOOTVID.DLL用于引导期间和发生蓝屏时的显示，KDCOM.DLL用于支持内核调试，CLFS.SYS是支持日志的内核模块，CI.DLL是用于检查模块的完整性的（Code Integrity）。

加载好系统模块后，WinLoad还需要加载引导类型（Boot Type）的设备驱动程序，在安装驱动程序时，每个驱动程序都会指定启动类型（Start Type），这个设置决定了驱动程序的加载时机，指定为引导类型的驱动程序是最先被加载的。

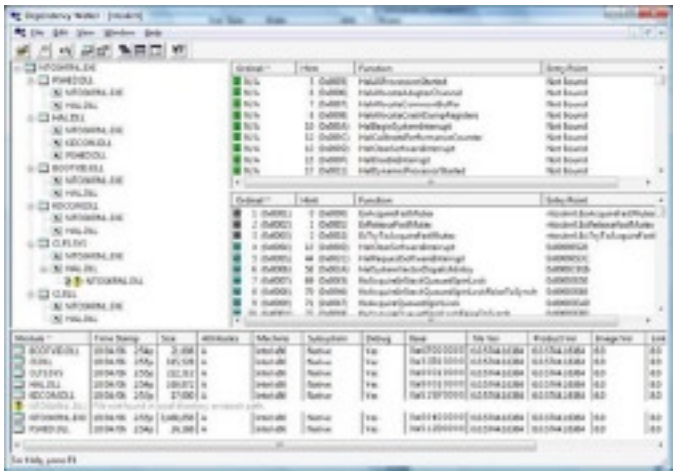


图2 使用DEPENDS工具观察NTOSKRNL.EXE所依赖的其它模块

如果在加载以上程序模块或者注册表的过程中找不到需要的文件或者在检查文件的完整性时发现异常，那么WinLoad便会提示错误而停止继续加载，我们在08年第11期中提到的问题便是与此有关的。当遇到这样的问题时，可以使用安装光盘引导，然后恢复丢失或者被破坏的文件。

完成模块加载后，WinLoad开始准备把执行权移交给内核，包括为内核准备新的GDT和IDT（OslArchpKernelSetupPhase0）和建立内存映射（OslBuildKernelMemoryMap）等。所有准备工作做完后，WinLoad调用OslArchTransferToKernel函数把供内核使用的GDT和IDT地址加载到CPU中，然后调用内核的入口函数，正式把控制权移交个内核。

启用调试选项

Windows Vista的BootMgr和WinLoad程序内部都集成了调试引擎，不管是Checked版本还是Free版本，对于Free版本，默认是禁止的，使用时需要开启，具体做法如下：

如果要启用BootMgr中的调试引擎，那么应该在一个具有管理员权限的控制台窗口中执行如下命令：

```
bcdedit /set {bootmgr} bootdebug on
bcdedit /set {bootmgr} debugtype serial
bcdedit /set {bootmgr} debugport 1
bcdedit /set {bootmgr} baudrate 115200
```

以上命令是使用串行口作为主机和目标机之间的通信方式，如果使用其它方式，那么应该设置对应的参数。

如果要启用WinLoad程序中的调试引擎，那么应该先找到它所对应的引导项的GUID值，然后执行如下命令：

bcdedit /set {GUID} bootdebug on

启用调试引擎并连接通信电缆后，在主机端运行WinDBG工具，便可以进行调试了，栈回溯、访问内存、访问寄存器等内核调试命令都可以像普通内核调试一样使用。

Windows Vista之前的情况

在Vista之前，NTLDR是Windows操作系统的加载程序。因为只有Checked版本的NTLDR才支持调试，所以如果要调试加载阶段的问题，应该先将NTLDR替换为Checked版本。DDK中通常包含有Checked版本的NTLDR程序。记住，在替换前，应该先去除NTLDR文件的系统、隐藏和只读属性，在更换后，要加上这些属性，否则的话引导扇区中的代码会报告NTLDR is missing错误，无法继续启动。

除了加载内核和引导类型的驱动程序外，NTLDR会调用NTDETECT.COM来做基本的硬件检查并搜集硬件信息。NTDETECT会把搜集到的信息存放到注册表中。如果找不到NTDETECT.COM，那么通常会直接重启，如果NTDETECT发现系统缺少必须的硬件或固件支持，比如ACPI支持，那么会显示因为硬件配置问题而无法启动，也就是我们上一期所提问的问题。对于这样的问题，可以尝试更改BIOS选项来解决，或者通过调试NTLDR来进一步定位错误原因。

恢复缺失文件

可以使用如下方法之一来尝试恢复丢失或者损坏的系统文件：

1） 启动时按F8，调出高级启动菜单，尝试选择Last Known Good Configuration（LKG）。

2） 启动时按F8，在高级启动菜单中选择安全模式（Safe Mode），如果成功启动后，那么可以尝试执行CHKDSK命令检查和修复磁盘，或者从安装光盘中恢复缺失的文件。

3） 使用Windows安装光盘引导，并记入到恢复控制台（Recovery Console）界面。对于Windows XP，在安装程序的主界面中按R键进入文本界面的恢复控制台，进入时输入管理员密码。对于Windows Vista，从安装光盘启动后，可以进入图形界面的系统恢复向导。如果是MBR或者引导分区损坏，那么Windows XP的恢复控制台中提供了FIXMBR和FIXBOOT命令。而Vista的恢复向导中包含了自动修复功能。

4） 如果系统硬盘的个数或者有所变化，那么可能是因为分区编号变化而导致系统无法找到文件，这时可以考虑恢复旧的磁盘和分区配置，或者启动到恢复控制台来修改系统的启动配置文件，对于Vista，需要修改BCD，对于Vista之前的系统，也就是修改BOOT.INI文件。

对于第11期的问题，天津的黄小非读者给出了非常好的答案，他的来信中给出了多种方法，包括使用控制台，使用Windows Preinstallation Environment（WinPE）以及修改BOOT.INI。其实Vista的恢复界面就是运行在WinPE中的。从黄小非的来信中，我们可以看出他的实践经验很丰富。■

本期优胜奖获得者

黄小非　天津市塘沽区渤海石油研究院计算机室

下一期问题

系统启动后很快出现蓝屏，其中含有STOP 0x0000007B INACCESSABLE_BOOT_DEVICE，哪些原因会导致这样的问题，该如何解决？

编者说明

- 投稿信箱：contest@csdn.net
- 解答提交时间，最好能早于当月15日。
- 联系方式写在一个单独的TXT文件里，包括：

1）姓名　2）工作单位或学校　3）电话联系方式　4）邮寄地址　5）E-mail地址

磨刀不误砍柴工
——IDE助你提高开发效率

如果说1964年的Dartmouth BASIC IDE算是第一个真正意义上的IDE，那么IDE已经陪伴我们走过了44年的风风雨雨。有人说IDE让程序员变懒了；有人说IDE让程序员疏远了编译和连接等操作。但是这并不能否定IDE为提升开发效率所作出的贡献。本期月度关注将带您走进Java IDE的世界，继续体验技术进步所为我们带来的便捷。

集成开发环境简史

■ 文 / 许舟平

如果说做软件就像盖高楼，那IDE（Integrated Development Environment）就像脚手架和升降机，帮助众多的程序员在软件的高楼上面爬上爬下。如果说做软件就像农民种地，那IDE就像锄头、爬犁、拖拉机，帮助程序员在软件的土地上耕作。如果说做软件就像一场没有硝烟的战争，那IDE就像士兵们手中的武器，帮助程序员在战争中或近身搏杀或远距离击杀。在软件发展数十年的时间里，作为软件工业中重要的生产工具——软件开发的集成开发环境IDE，一直以来都在随着软件产业的发展而进行着自我演变。

IDE的石器时代

在早期的电子管、晶体管计算机时代，计算机软件仅仅通过简单的机器语言和汇编语言，存在于各个研究机构和国家实验室中。如果程序员想要编译一段代码，必须先要走流程图，再撰写表格，然后打卡，最后才开始执行。如果说流程图－表格－打卡也可以被称为一种最原始的集成开发环境的话，那就是IDE的旧石器时代。

在以集成电路为主要技术的第三代计算机出现之后，计算机软件得到了飞速的发展，计算机软件的需求也随之日趋复杂。从前靠打卡来执行代码的方式在今天看来如同钻木取火一般。软件的编写需要更为可靠的工具来完成，代码的编译工作和调试工作也需要适应越来越多的软件开发者在编写软件时的各种需要，就像在远古时代从独自狩猎演变成群体狩猎一样，狩猎的武器由经过打磨好的锋利石器代替了随手从地上捡起的石块。一个全新的IDE工具产生了，我们可以称之为IDE的新石器时代。

新石器时代早期的IDE工具其实功能非常简单，仅仅是由一个源代码编辑器，一个程序编译器和一个调试器组成。由于对程序编译器的运行条件的要求不同，那时的IDE大多只能在一种平台上为一种计算机语言工作。第一个拥有新石器时代IDE工具的计算机语言是Basic语言，那时Basic IDE中的“IDE”并不是今天所说的“Integrated Development Environment”，而是被称为“Integrated Design Environment”，可能Design更能够代表当时的程序员编写代码时感受到的是艺术性的设计而不是机械式的开发吧。

第一个Basic语言的IDE是作为Dartmouth Time Sharing System（DTSS）的一部分推出的。对于分时操作系统有所了解的程序员应该都知道DTSS的大名。DTSS是由Dartmouth College在1963年推出的世界上第一个大规模分时操作系统，DTSS创造了软件史上很多个第一，其中就包括了第一个可以直接在主机或终端机上开发的IDE工具。Dartmouth BASIC IDE采取命令列的方式，没有像今天其他的IDE工具那样具有选单和图形化的功能。它只拥有非常简单的new/old/list/save/run这几个功能。

IDE的青铜时代

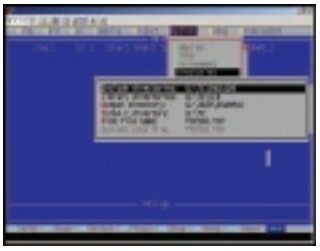
在经历过软件危机之后，“面向过

程”的编程方法第一次将程序员的思想带入了一个以事件为中心的软件工程时代。此时的计算机语言已经如雨后春笋般出现在各个计算机应用的领域，而UNIX的出现更是将计算机操作系统带入了一个全新的世界。作为程序员最主要的生产工具——IDE注定要在这个时代经历一场大变革。在这场变革中，有这样两个公司曾经生死相斗、短兵相接，那就是Microsoft公司和Borland公司。

早年Borland公司推出的Pascal海报

上世纪80年代初，PC机的普及将软件开发带进了一个前所未有的境地。在Turbo Pascal 1.0出现之前的IDE工具，虽然都具备源代码编辑器、程序编译器和调试器等这些IDE石器时代就具有的功能，并从人机交互的体验上有所提高，但是如果当时一个程序员想要通过IDE完成软件开发，在编写完代码之后程序员必须要退出源代码编辑器返回操作系统，如UNIX或者DOS，然后再运行程序编译器进行代码编译。要是一次通过还好办，如果赶上水平差点的程序员或者是初学者，编译时遇到了错误，程序员将不得不自己记下错误的行数，再次打开源代码编辑器进行修改保存，然后重新返回操作系统进行编译。如此周而复始的直到代码可以顺利通过。

在Turbo Pascal推出4年之后的1987年，Borland公司继续着自己在IDE市场上的攻城略地，首次推出Turbo C 1.0产品。Turbo C使用了全然一新的集成开发环境，包括了一系列下拉式菜单，将文本编辑、程序编译、连接以及程序运行一体化，大大方便了C语言的开发工作。可以说Borland公司是IDE青铜时代的王者。Turbo系列的集成开发环境改变了传统软件开发的习惯，让越来越多的程序员可以迅速的掌握软件开发的技巧，并极大地提高了软件开发的效率。


还记得大学机房中那个熟悉的蓝色界面吗？

IDE的铁器时代

从上世纪80年代中后期开始，数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理等方面的应用层出不穷，而基于传统面向过程的开发模式受到了很大挑战。许多程序员开始慢慢采用将自己常用的函数进行封装、继承甚至于抽象的设计方法来进行软件开发，从而将软件工程由面向过程编程逐渐演变为面向对象编程。作为面向对象的基本特征：对象的唯一性；数据结构(属性)和行为(操作)的抽象性；子类自动共享父类数据结构和方法的继承性；相同的操作或函数、过程作用于多种类型的对象上的多态性在软件设计中大量使用。因此，作为软件设计的工具——IDE，也为此发生了翻天覆地的变化，IDE铁器时代到来的标志就是可视化集成开发环境的出现，例如微软的Visual Basic，Borland的Delphi等。而今天我们所熟悉的Eclipse、Netbeans、JDeveloper、IntelliJ IDEA、JBuilder、Visual Studio等等，都是踏着前人的脚印一步步的走来。

可视化集成开发环境的特点就是“控件组装”，很多控件都是由程序员自己像画图一样构建出来。可视化集成开发环境帮助今天的程序员解决了很多例行的、标准化的代码，让程序员从繁杂的事务性编程工作尽可能的抽出来，去完成更具有创造性的开发工作，从而加快项目的开发速度，提高每一个程序员工作效率。

让我们把时间回到1991年，眼看着Borland公司的IDE产品越来越强大，不甘心在Pascal上失败的微软公司推出了Visual Basic 1.0版。其实用今天的眼光来看，VB1.0的功能实在是太弱了，但在当时它是第一个“可视”的集成开发环境。VB的出现让基于DOS/Windows平台的开发者都兴奋不已，纷纷尝试在VB的IDE平台上进行针对当时还是Windows3.x平台的应用开发。

而在这个期间，Borland公司也在不断地进行改进，于1991年将Turbo C++作了更新，推出了新一代产品Borland C++。之后Borland公司又购买了White Water的C/C++ Framework，把它更名为OWL（Object Windows Library），并且融入到当时最新的Borland C/C++ 3.1产品中。由于Borland OWL 1.0比当时微软的MFC 1.0封装得更为完整且好用，还加入Resource Workshop可视化能力，使得Borland C/C++ 3.1一度占据了C/C++集成开发环境的半壁江山。而作为帮助Borland打败微软的青铜时代利器——Turbo Pascal，也被Borland公司开发出了可以在Windows平台运行的可视化IDE版本，那就是大名鼎鼎的Delphi，这一年是1995年。也就是在1995年，一个能够改变世界的计算机语言诞生了，那就是Java。

让我们来一起回顾一下Java语言在公元1995年的发展历史：

- 1995年5月23日Sun在SunWorld'95上正式发布Java和Hot Java浏览器。
- 1995年8月Netscape获得

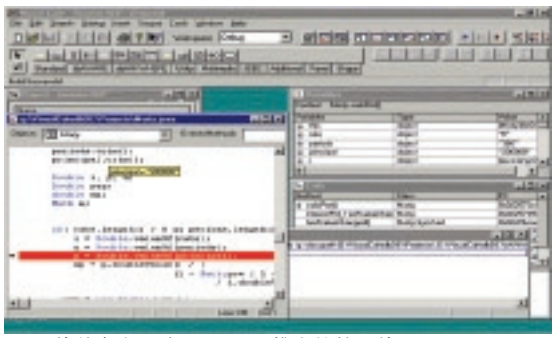
Java许可证。

● 1995年9月Sun宣布将提供Java开发工具。

● 1995年10月Oracle公司、Borland公司获得Java许可证。

● 1995年12月SGI公司、Adobe公司、IBM公司、AT&T公司、Intel公司获得Java许可证。Sun和Netscape发布JavaScript——一种基于Java语言的脚本语言，可供非编程员使用。

在计算机语言的发展史上，很少有一种语言像Java一样一经推出就受到如此的热捧。一种软件的生命力就在于创新，对于创造软件代码的集成开发环境更是在于此。在今天，有谁能够想到世界上第一个基于Java的可视化开发工具是由安全巨头Symantec公司提供的呢？没错！1996年10月，Symantec正式推出了Visual Café 1.0。之后不久，在众多软件开发者的期待下，Java的缔造者Sun公司推出了自己的Java开发工具Sun Workshop。其实Sun的Java Workshop从一开始推出就是一个错误。在服务器和工作站上拥有丰富经验的Sun对于在PC上的开发者的使用习惯和开发需要了解甚少。Java Workshop只适合使用在昂贵的Sun工作站和服务器上，在普通的PC机上执行缓慢而笨拙。Sun在Workshop上的失败让这颗太阳黯然失色，从而老老实实的转向去做Java语言研究和JDK类库的开发工作。这也为后来Borland公司的JBuilder和IBM的Eclipse留下了巨大的发展空间，当Sun回过头再想来发展自己的IDE时，就推出了今天我们看到相比于Eclipse和JBuilder依然年少的Netbeans。

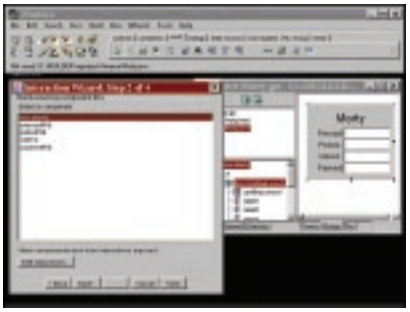


软件安全巨头Symantec推出的第一款Java IDE，时间是公元1996年10月

由于微软对于Java本身的态度问题和Visual J++的失败，对于Java IDE早期的争夺就集中在Borland公司和IBM公司之间。Borland公司将早年从事过Borland C/C++、Delphi和Paradox（一款DOS时代伟大的数据库产品，足以和今天的Oracle媲美）开发的许多精兵强将都调集到JBuilder项目组中。在Borland开发JBuilder之时，由于Sun公司的Java尚没有完整的组件架构，也没有完整的Java可视化组件，因此IDE组件开发经验丰富的Borland公司决定先自行开发一套Java组件，以便让JBuilder拥有最好的组件开发能力。这就是JBCL（JavaBeans Component Library）的由来，而JBCL的架构稍后也成为Sun公司制定JavaBean的基础技术。

对于IBM，早期IDE产品Visual Age For Java的停滞不前让JBuilder在Java IDE的市场上独领风骚。这样当JBuilder 4推出的时候Borland公司已经占领了Java IDE 50%的市场。由于JBuilder成功的竞争策略，让JBuilder的最大竞争对手IBM的Visual Age For Java和其他的Java IDE厂商都愈显吃力。当IDE市场已经养活不了这么多软件IDE厂商之时，Java的IDE已经由纯开发工具向着集成面向对象的大型系统开发工具的方向进行着转变。开发Case以及UML工具与IDE的集成就像是铁器时代中炼钢术的产生。Rational和TogetherSoft对于OO和UML，IBM和BEA对于EJB，还有开放源代码的时代浪潮，都在不断的影响着使用IDE的软件开发者。IBM适时的将Visual Age For Java灌入有着开源名号的Eclipse名下，并最终于2001年底成立了Eclipse协会和eclipse.org网站。

Eclipse是一个非常成功的开源项目，Eclipse通过一个非常方便的插件组件系统来构建每一个开发者所需要的开发环境。不仅如此，Eclipse还提供了一套插件开发环境（Plug-in Development Environment，PDE），主要针对希望扩展Eclipse功能的软件开发人员来进行开发，并且允许他们构建与Eclipse环境无缝集成的工具。Eclipse的出现将IDE的工具带入了一个全新的领域，开源、自由、社区这些程序员与生俱来的基因被打造在更多的Java IDE中。



赫赫有名的JBuilder 1.0时的模样

在今天我们的IDE兵器库中，无论是微软的Visual Studio，还是曾经和JBuilder同源的Oracle JDeveloper；无论是打着开源旗号的Sun Netbeans，还是收取费用的IntelliJ IDEA。这些大大小小、或开源或商业的IDE产品正帮助我们的程序员高效的在软件土地上耕作，洒下的是代码，收获的是软件。未来的IDE工具将会在智能化，简约化，高效化方面跟随着计算机语言的发展而进一步完善，让我们一起来见证。■

作者简介

许舟平，长期关注软件技术发展，在敏捷开发方法和项目管理等方面，具有多年实践经验。目前就职于一家美国公司，从事软件安全方面研究，为互联网、电信、政府和金融等行业提供安全策略。笃信以人为本，关注软件，专注技术。

■ 责任编辑：李雨来（liyl@csdn.net）

使用 Mylyn 提高开发效率

■ 文 / 秋实

在文章开始之前，我想请使用Eclipse作为自己开发环境的Java开发人员思考以下几个问题：

1、你是否觉得在开发过程中花费了很多时间在文件树中寻找文件？

2、当你在下班的时候，如果某个任务还没有完成，你是不是特别不愿意关闭集成开发环境（IDE），以便第二天能继续上一天的工作？

3、当你正在忙于手头的某个工作的时候，突然有个优先级更高的任务需要你处理，导致你不得不切换工作空间或者需要关闭当前任务的相关文件，等你把那个优先级更高的任务处理完之后，你发现你不得不花时间再一次找到之前任务所需的文件，以便继续手头的工作，这个问题是否让你觉得很烦恼？

如果你对以上问题的答案是“是”的话，那么我强烈推荐你继续读完这篇文章，因为Mylyn能够帮你解决以上问题。

让我们先来看看这些问题的根源吧，软件随着自身的发展体积越来越庞大，当我们开发软件系统的时候，由于系统的复杂性，我们需要通过模块化来有效地组织代码。所以一个大型的应用往往由多个模块组成，随之程序文件也往往是成百上千。而人本身处理信息能力的局限性，不可能对所有信息进行处理，虽然集成开发环境（IDE）通过结构化视图，高级搜索等功能来帮助开发人员快速定位需要的信息。但它缺乏某种途径，来有效地记录用户的工作任务和任务所需资源（源代码，文档）之间的关系，并根据用户当前所做的任务来有效地过滤与任务无关的信息。因为对于某个任务来说，与之相关的资源只局限于很小的一部分。所以有时候当我们在某个任务被打断的时候，我们会把当前任务所涉及的资源名称进行记录，以便之后可以继续进行。这种方式比较原始，也无法进行有效地过滤。正是这个原因，严重限制了开发人员并发处理任务的能力。设想一下，如果你同时需要处理10个任务，每个任务都关联不同资源，而你需要在不同任务之间频繁切换，那是一件多么痛苦的事情！而通过Mylyn的自动化上下文管理，这些问题会得以轻松解决。

让我们先来了解一下Mylyn是什么吧。简单来说，Mylyn是一个与Eclipse高度整合的上下文管理及任务管理工具。

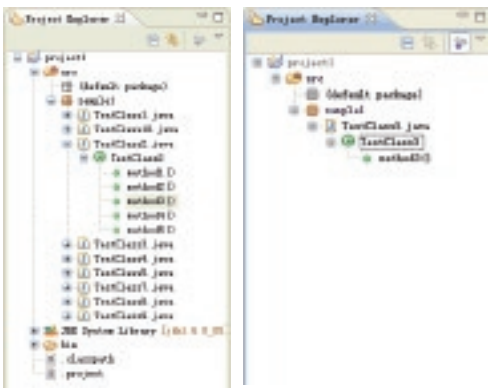
Mylyn的自动上下文管理

首先阐述Mylyn中的两个基本概念。

任务：开发人员需要做的某件事，比如说开发一个新功能，修复一个bug，对一部分代码进行重构等等。

任务上下文：这个任务执行时所涉及的资源，比如说源代码，配置文件，设计文档等等。

自动上下文管理是Mylyn的最大亮点，它几乎不需要你做额外的工作，就可以帮助你将任务及任务的上下文进行有效关联，并在Eclipse中对与任务无关的信息进行过滤。让我们来看个例子吧，假设你要编辑project1下TestClass2中的method3方法，下图分别是你没有使用和使用Mylyn后所看到的Project Explorer视图：

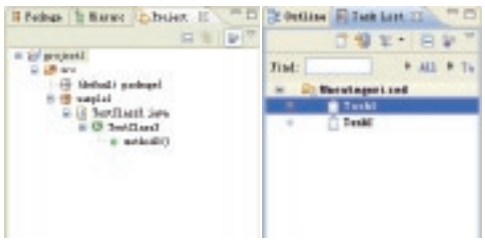


（未使用Mylyn）　　（使用Mylyn）

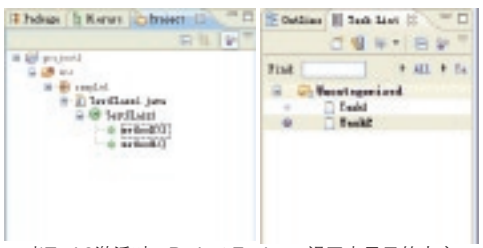
试想一下，如果你的项目中有上千个文件，而你只需要编辑其中的一两个文件的话，上图又将是怎么样的一幅情景。有了Mylyn的帮助，将为你免去在文件树中寻找文件的烦恼。而要做到这一点，你只需要简单的定义一个任务即可。

打开Task List视图，在视图上单击鼠标右键，选择“New->Task”即可创建一个任务，Repository选择Local Repository即可（其他的Repository我们会在Mylyn和第三方系统集成的时候进行介绍），输入任务相应信息，任务的创建就完成了。该任务就会出现在Task List视图中。在该任务上右键单击，选择“Activate”即可激活该任务。在任务激活之后，用户在Eclipse中操作的文件都会被添

加到该任务的上下文中，对于Java文件，能过滤到方法一级，只有在任务的上下文中的方法才会显示在相应的视图上。我们可以通过创建和激活不同的任务，由Mylyn来自动的关联该任务与之相关的上下文。例如我们还有一项任务，需要编辑TestClass1的method2和method4方法，当我们在不同任务之间切换时，只需要激活不同任务，与任务相关的上下文就会自动打开。下面两图是当激活不同任务时，Project Explorer视图展示出与任务相关的内容：



当Task1激活时，Project Explorer 视图中显示的内容



当Task2激活时，Project Explorer 视图中显示的内容

Mylyn的任务管理

在Mylyn的Task List视图中，还可以完成对任务的管理。通过Mylyn的任务管理，我们可以管理工作周内所有相关的任务。对于很多开发人员来说，需要在一个给定的工作周内跟踪和调度的任务的范围包括：

1、为您正在开发的产品解决产品缺陷并分析特性。

2、由您的同事完成的任务，这些同事包括您的搭档、上司或下属。

3、针对您所使用的框架、API和软件的bug报告。

4、个人待办事项和提醒。

将所有任务集成到一个视图中，可以使它们更易于管理，因为只需查看一个位置就知道接下来该做什么。为了进一步简化任务管理，Mylyn隐式包含了一些公认的任务管理最佳实践，例如调度和延迟任务，并且还包括了XP式开发的即时性和适应性。Mylyn的任务管理工具可以使您轻松地适应一周内的任务变化需求，而不会丢失对长期优先级的跟踪。

Mylyn提供两种用于调度的日期：预定日期（scheduled date）和到期日期（due date）。

任务的预定日期是根据个人日程安排的可变日期，可以根据优先级的变化而推迟。它定义你计划开始处理任务的时间。当任务到达预定日期时，任务变成红色，以指示你应该开始处理它或者考虑推迟它的起始日期。预定在今天执行的任务会变成蓝色而不是红色，并且在工作日的最后进行安排，避免重新安排整个工作日的任务。

任务的到期日期是固定的日期：任务必须完成的日期。这个日期常常与外部的约束（例如最后期限）相关。到期日期的临近是通过任务图标上的一个小钟指示的，这个小钟在到期日期之前为蓝色，在到期日期之后为红色。

Mylyn还提供了丰富的搜索，排序，过滤，分组等操作，以帮助用户快速定位到当前需要完成的工作。

Mylyn和第三方系统集成

Mylyn不仅能创建本地任务，还可以和第三方的系统集成，如Bugzilla、Jira、Xplanner等等。具体支持哪些第三方系统，用户可以通过访问以下URL来获取最新的信息：（http://wiki.eclipse.org/index.php/Mylyn_Extensions#Task_Repository_Connectors）■

作者简介

秋实，软件工程师。目前在中国北京IBM CSDL从事J2EE，SOA的方面的研究和开发。

■ 责任编辑：李雨来（liyl@csdn.net）

在目前的Java IDE市场已经形成三足鼎力：Eclipse、IntelliJ IDEA和NetBeans，对于国内很多开发人员来说，Eclipse众所周知，NetBeans次之，IntelliJ IDEA确很少被人了解。IDEA不被人了解的主要原因是其开发公司不大，而且中文文档比较少，宣传也很少。IntelliJ IDEA号称最智能的Java开发工具，通过这篇文章，我们将向大家介绍这款智能Java开发工具以及如何利用IDEA提升你的开发效率。

从编辑器起家

IntelliJ IDEA的最初版本就是一款Java代码编辑器，着重于代码编辑的功能：智能代码提示、代码生成、代码风格管理、导航和搜索等。虽然这些都是IDE的基础功能，但IDEA在最早版本就努力实现这些基础功能，并且力图做到最优。由于这些基础功能的实现，IDEA使得编辑Java代码变得非常简单。除此之外，Java主要凸显了重构、Ant整合和JUnit支持，与版本控制系统的整合，一举IDEA 2.5成为当年最佳IDE工具。随着IDEA的版本升级，这些基础特性得到不断地升级，由最初的基本功能逐步向智能化方向发展。

重构可以说是各个编程语言都会涉及到，不只是Java，由于程序的复杂性不断增加，各种文件类型要配合才能成为一个完整的应用，如一个Web应用就涉及到Java、JSP、XML、CSS等等，而且各个类型都存在相互引用的例子，如web.xml就会牵涉到对Java类名的引用。重构也由单一文件类型逐步向多种文件类型辐射，力求代码的结构完整。如你给一个Java类重命名，不只是Java代码中的引用会改变，其他如JSP、web.xml、Spring的xml配置等等，这些都应该做相应的更改，不会有任何遗漏，避免了重构不到位的情况。这种多语言、多类型的交叉重构，让IDEA在重构领域一直遥遥领先。

IntelliJ IDEA——开发人员利器

■ 文 / 陈立兵

编辑图形化：伴随着IDEA升级，编辑器逐步向图形化发展，如web.xml的编辑已经可以图形化操作；Swing开发提供了图形化支持；HTML编辑后可以实时预览；在最新的IDEA 8.0中还添加了UML功能，实现图形和代码的完全同步。这些都是编辑器转变的过程——朝着更便捷的方向发展。顺便插一句，IDEA最早的核心开发人员都来自Together（一款著名UML工具），我们有理由相信IDEA会在图形和UML方面做得更好。

编辑智能化：很多人可能都看过极限编程方面的书籍，在IDEA中，它不仅仅是IDE，同时也是你的伙伴，这种特性主要通过代码审查(Code Inspection）和意图动作(Intention Action）实现。代码审查主要是发现你代码中潜在的bug（不是语法错误），很多人可能都知道FindBug，但是IDEA的审查机制是实时的，你每输入一个字符，IDEA都会帮你判断，什么地方的变量声明但没有使用、空指针判断、重复代码等等，就向你的助手实时在提醒你。意图操作则是根据上下文环境，来揣测你将要进行的操作，如将光标置于Java接口名称上，IDEA会提示你是否进行接口实现；当你在web.xml中声明一个不存在的Servlet类名，IDEA会提示是否进行Servlet创建等等。而代码审查和意图操作都提供了相关的修复方案，当发现问题后，你只需要一次鼠标点击或按一下回车键就可以完成，完全是一个合格的伙伴。在IDEA 8.0中，已经有800多条审查规则，和200多项意图操作，足以让你的代码质量得到很大提高。

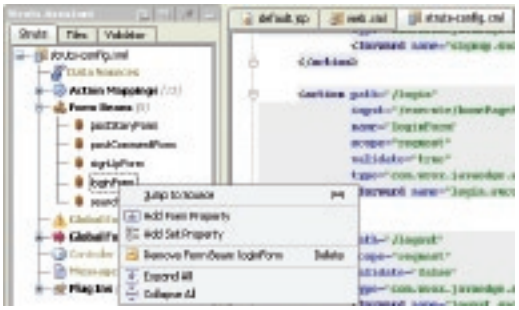
项目结构：当越来越多的文件加入到IDE工具中，就需要对其进行整理和规划。IDEA提供了完善的项目结构和配置。项目(project)统领整个工作区域，然后各个模块（module）负责具体功能块的实现，在此基础上IDEA添加了构面（facet）支持，可以将多个特性添加模块上，如Web Facet、Hibernate Facet等等，这样模块就具有更多的特性，同时也可以快速了解模块的特性。IDEA很早就添加了对TDD支持，将Unit Test纳为默认支持，而且目录结构方面也朝这个方向发展（Test Source就是这个特性的呈现）。在和Maven整合后，IDEA可以完全遵循Maven的项目结构，项目管理也标准化很多。

Web开发

Web应用可以说是目前开发的主流，IDEA在J2EE 1.2发布的时候就添加对其支持，这一点在IDEA 3.0就得到了体现。让我们看一下Web开发涉及的技术，然后再看一下IDEA是否提供相应的支持。

应用服务器整合：首先我们希望IDE能整合应用服务器，将IDE和应用服务器整合后，可以进行应用服务器相关的配置，调试以及快速部署。目前IDEA整合了Tomcat、Resin、JBoss、Weblogic等，所以在IDEA下你可以快速启动这些应用服务器并进行应用调试，打包也非常便捷。

Web框架支持：就Java的Web应用而言，我们通常需要选择一款Web框架，这样会给开发带来便捷，同时应用结构也会比较规范。回到Java的Web框架选择上，目前Java的Web框架比比皆是，而IDEA对主流的Web框架都提供了支持，如Struts 1.0和2.0、JBoss Seam、Spring MVC、JSF等。如果你选择的框架不在其列，扩展也非常简单，如Stripes plugin、Wicket plugin等，这些框架通过plugin机制很容易实现。



Struts 1.0框架在IDEA下的支持效果图

页面端技术：Web应用的前端是非常重要的，首先IDEA提供HTML/XTHTML/CSS的支持，包括代码提示、代码校验等，在此基础上，IDEA支持JSP编辑和JSP的相关规范：如Taglib、JSTL、EL等，而且各种语言的混合处理非常到位。IDEA 8.0提供了浏览器整合，Javascript Debug，HTML实时预览等，这些对Web开发起到了很好的辅助作用。由于Web开发牵涉到的知识和资源比较多，因此IDEA做了全面的整合，这些资源都

可以做到相互交叉使用和相互嵌套，如Javascript和DOM整合，CSS和HTML整合等。按照水桶理论，IDEA不但每一项功能过硬，而且整合后能很好地成为一体，从而避免短板效应。由此FreeMarker/Velocity等模板语言加入后，在模板代码中，Java、HTML、Javascript语言均可以得到支持，这些都好像原生的一样，不会有任何功能缺失。

Javascript：这里将Javascript单独提出来，主要是Javascript在Web开发中扮演的角色越来越重要，IDEA支持各种Javascript框架，如Prototypes, JQuery、MooTools、YUI和ExtJS等，还有就是Google的GWT等RIA框架。对于Javascript中重要的JSON数据结构，IDEA也提供内置支持。有了这些，IDEA还提供Javascript Debugger，这样在IDEA下可以轻松调试Javascript应用。

IDEA对Javascript的支持

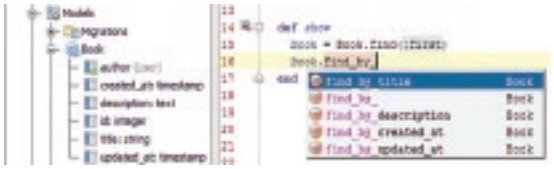
其他：Web应用中可能还会牵涉到其他技术，如Web Service，IDEA也提供了默认支持，可以轻松创建Web Service应用，同时第三方plugin，如soapUI、RESTClient，对Web Service调试提供了便捷地服务。还有一些就是IDEA提供了很多关于Web的Open API，这些都可以让开发Web相关的第三方插件变得更加简单。

框架和插件机制

随着Java的发展，越来越多的框架被开发出来，如Struts，Hibernate和Spring等，这个时候IDEA推出Plugin机制，通过对Open API的扩展，来实现自定义功能。IDEA Open API除了提供Plugin的框架结构，同时也提供了大量的服务，这样做功能扩展就比较容易，如IDEA提供了很多的工具方法和组件，将IDEA本身的特性逐渐插件化，我们只需要扩展相关的插件，使用相关的组件和方法，就可以实现非常实用的功能。如基于XML配置的框架，通过扩展XML的编辑功能，配置编写就非常简单了。在Open API的基础上，大量的插件被开发出来，涉及IDEA的方方面面，你可以从http://plugins.intellij.net查询到这些插件。虽然有大量的插件，但是IDEA官方致力于更多的底层功能和关键性插件开发，如VCS相关的插件：Database Plugin、Git Plugin、Maven Plugin、代码覆盖Plugin、IntelliLang等。这些插件的质量和都比较高，而且全部开源，为其他的插件的开发提供了有力地指导。此外IDEA提供了J2ME开发的支持，这个也是不少手机Java游戏开发人员值得关注的。

平台，更多语言：Groovy, Ruby, Python

随着IDEA 8.0推出，IDEA不仅仅是一个IDE工具，而是向平台化转变，基于这个平台，可以方便扩展各种功能。和Eclipse平台不一样的是，IDEA平台主要关注底层服务，提供了很多的基础服务和优良插件。因此基于这个平台构建工具将更加简单，只需要扩展一下平台，添加一些相关的Plugin，就可以组成特定语言的IDE，这里典型的例子就是RubyMine工具。RubyMine包含IDEA的核心平台，然后加入VCS相关的插件、Web插件(HTML/CSS/Javascript等)，再结合Ruby Plugin，整个Ruby IDE就搭建完成。回头看一下RubyMine的特性，Rails支持，各种Web开发特性，如HTML编辑、Javascript支持等，完全是一个特性十足Ruby IDE。目前JetBrains所致力开发的主要是Ruby、Groovy、Scala和Python等独立语言插件，而且这三个插件在各自的语言领域已经得到认可。在IDEA 8.0还添加了对Flex支持，也是对ActionScript语言的Plugin实现，而且目前的功能已经非常完善。为了配合多种语言，在IDEA 8.0中提供了对SQL的原生支持，在数据源的配合下，可以对.sql文件给予表名、字段名、函数等的智能自动提示；并且还能对Hibernate的HSQL、iBatis的嵌入SQL、JDBC的字符串SQL提供跳转、高亮、审查等的支持。



IDEA对Rails的支持

展望

现今IDE的工具竞争可以说是相当激烈，伴随着每一次升级，IDE的工具都再度变得更加庞大，功能越来越完善，在将来的某一天IDEA是否会闯入UML领域还是个未知数。很多人会有这样的疑问，工具大了是不是不敏捷啦？我想不是的。首先工具应该能定制，也就是不需要的功能可以取消掉，IDEA采用插件机制，你不需要的插件可以停掉，这样IDE会整洁很多；其次在工具变大的同时，我们希望工具能深挖掘基本特性，比较常用的功能占80%的操作时间，基本功能的一点点改进都会给每一个开发人员带来效率地提高。最后希望IDE工具能够快速响应某些新型技术，让开发人员能快速体验到新技术带来的便捷。■

作者简介

陈立兵，IntelliJ IDEA(http://www.intellij.org.cn)中文爱好者站点站长，主要致力于IntelliJ IDEA研究，开发过不少IDEA插件，目前就职于淘宝网。

■ 责任编辑：李雨来（liyl@csdn.net）

新产品 & 工具

OpenCL 1.0

长久以来，业界就一直试图要把强大的绘图处理器运算能力运用在一般运算上，这个概念也就是所谓的GPGPU（general-purpose computation on GPUs，通用GPU运算），而OpenCL可望实现这样的目标。OpenCL（Open Computing Language，开放运算语言）是一个开放而免授权的跨平台编程语言标准，专为当今个人电脑、服务器以及手持和嵌入式设备的并行处理所设计。OpenCL可以让绘图处理器在电脑中不仅仅只用于绘图等相关应用上，而且能执行各种不同的应用程序。Khronos Group最近宣布通过并公布OpenCL 1.0规范。OpenCL程式语言的设计，也是为了要让软体开发人员能够充分组合利用各种不同的多核心处理器、绘图处理器（GPU）、Cell架构，及其他平行处理器（如DSP）的性能。让绘图处理器在电脑中不仅仅只用于绘图等相关应用上，而能执行各种不同的应用程序。

Oxite

近日微软推出了一个开放源代码的博客(blogging)平台Oxite，Oxite与标准相容、可管理延伸内容的系统，用意在支持博客或更大型的网站。使用者可在网站上建立、编辑一系列的页面，把定制化的HTML语法加入网页，并支持在单一网站上设立多重博客。Oxite也可与微软的多款开发者软件整合，例如ASP.Net MVC、Visual Studio Team Suite以及Background Services Architecture。

用友华表 E-Cell 业务设计与运行系统

2008年11月28日，用友集团旗下的北京用友华表软件技术有限公司正式发布基于类Excel环境的业务设计运行平台的“用友E-Cell业务设计与运行系统”。用友E-Cell系统是在基于用友三大核心技术之一用友Cell组件、插件上自主开发面的。系统分为设计端和运行端，设计端可以让用户在熟悉的类Excel环境下通过“表单模型+流程模型+报表模型”的设计，实现业务系统搭建。

TortoiseGit 0.1.0.0 preview

TortoiseSVN是windows平台下非常好用的一个Subversion客户端，类似的也有TortoiseCVS。作为Git的用户，可能很多时候都会对于Git的图形化客户端感到失望，数量很少并且大多数都并不是那么的易用、强大。而现在，Tortoise系列放出了针对Git的版本，虽然还是一个很早期的预览版，但是已经能够看到其中的一些特性和功能了。

Linux Mint 6 Felicia 正式版

Linux Mint是一份基于Ubuntu的发行版本，其目标是提供一份完整意义上的即刻可用的体验。而这会通过包含浏览器插件、多媒体编码解码器、DVD播放支持、 Java及其他组件来实现。它也增加了一套定制桌面及各种菜单，一些独特的配置工具，以及一份基于Web的软件包安装界面。Linux Mint团队刚刚发布了基于代号为 Felicia 的 Linux Mint 6 正式版，该版本基于Ubuntu 8.10，采用Linux 2.6.27 内核、Gnome 2.24桌面环境以及Xorg 7.4 。

OpenSolaris 2008.11

OpenSolaris 2008.11 是 OpenSolaris 操作系统的最新版本。新版包括GNOME2.24、ZFS文件系统、全新的打印管理器、桌面搜索、新的包管理器、Netbeans 6.5、DTrace、Firefox 3、OpenOffice 3、Gobby、休眠功能、基于Mozilla 的 XULRunner的音乐播放器Songbird等等，此外还改进了安装、网络管理、硬件支持、Web栈等。

Google Picasa 3.1

Google Picasa作为一款在线图片应用的客户端，其价值不仅仅局限在网络应用方面。Picasa可以帮助你在计算机上迅速找到、修改、管理和共享所有图片。本次发布的新版本支持多达38种语言，当然也包括中文。此外还包括了多种改进。

VirtualBox 2.1.0

VirtualBox是德国一家软件公司InnoTek所开发的虚拟系统软件，目前已经被Sun所收购，并成为了一款开源软件。VMWare Workstation不错，但可惜是收费的。在免费的虚拟机这个层面上来说，相对于VMWare Server/Player和KVM等，VirtualBox最大的优势在于跨平台、开源，同时性能也不错。新版本做出的改进包括：

- 支持MacOS X上的硬件虚拟化指令（VT-x and AMD-V）
- 支持在32位的主机上虚拟64位客户机
- 通过OpenGL支持3D加速
- 支持LsiLogic和BusLogic SCCI控制器
- 支持 VMware的硬盘镜像文件 VMDK和VHD
- 新的NAT引擎和网络接口

Linux 下 64 位 Java 插件

Adobe于近期推出了64位版本的Flash For Linux插件，而现在Sun Microsystems也于日前推出了Linux下64位版本的Java插件，对于Linux用户来说，切换到64位操作系统更加的平滑了。该插件兼容64位版本网页浏览器，此次64位Java插件作为Java SE 6 update 12 Build 02的一部分，同样适用于Linux与Windows下，而Solaris将会在有64位版本的Firefox时适时推出相应的64位 Java插件。

DevPartner Studio 9.0

Compuware的DevPartner Studio 9是一款功能强大的代码质量保障软件。DevPartner通过在每次编译时扫描应用源码，并通过内建的200多条安全规则对代码进行检查，从而帮助开发人员提高代码质量。此外，DevPartner还能提供集成的报告、生成代码质量报告，便于管理人员和团队负责人通过Web浏览器进行检查。

Mono 2.0

由Novell所支持的Mono项目是一个开源的跨平台.Net开发框架。日前发布了 Mono 2.0版本，提供了在Linux上开发和运行.Net程序的必备环境，与Microsoft .Net Framework 2.0 兼容。根据IDC的统计，目前有超过50%的IT决策者、开发人员和架构师使用Microsoft .Net Framework开发和运行一些关键性应用。通过Mono 2.0，开发人员可以平滑的将其现有的技能应用到多种平台之上，包括Linux、Solaris、Unix和Mac OS X。

业界新闻

软件过程管理“信息化”时代 ——2008 QONE4.0 产品发布会成功举办

11月28日，中科方德软件有限公司（基础软件国家工程研究中心）成功举办了软件过程管理“信息化”时代——2008 QONE4.0产品发布会。“方德软件过程管理平台Qone4.0”依据ISO9000、CMMI等标准的量化管理思想和管理方法，围绕软件过程的四类活动：即过程管理活动、项目管理活动、软件工程活动和软件支持活动。是一个覆盖软件过程资产、软件过程数据和软件过程文档的软件质量管理框架。

2008 Wind River 中国开发者区域大会召开

12月8-12日，设备软件优化厂商风河系统公司举办的2008 Wind River中国开发者区域大会，在国内几大重要城市巡回举行。本次大会的主题主要围绕着多核处理器虚拟化技术，Multicore Offloading(多核加速)技术的最新进展情况展开，并针对这些技术在各行业应用中的不同特性及需求进行深入讨论，消费电子、网络通信、航空航天与国防以及工业控制都是此次会议的重点领域。

Adobe Creative Suite 4 中文版用户大会成功举办

12月2日，Adobe公司在北京、上海、广州三地同步举办了主题为“登峰造极之径”的Adobe Creative Suite 4中文版用户大会。这次大会上，Adobe公司进行了迄今为止最大规模的软件版本发布。一千多名来自各行业的设计和开发人员、合作伙伴在大会上体验了Adobe Creative Suite 4中文版新产品。同时，上海和广州用户也可以通过Adobe的网络媒体技术，在本地会场观看讲师对新产品功能与特性的介绍。Adobe公司全球副总裁、创意解决方案部门设计和网络业务总经理David Burkett先生做了题为“新媒体、新动力”的主题演讲，并表示新产品提供的功能，将实现跨越不同媒体而展现一致的设计构想，用以满足人们从各种途径获取精彩体验的需求。

微软 Windows 硬件工程大会 WinHEC 2008 在北京揭幕

12月3日，亚洲规模最大的微软Windows硬件工程大会WinHEC 2008在北京揭幕。微软在大会上展示了其下一代操作系统Windows 7，并发布了Windows Home Server的中文版以及.NET Micro Framework 3.0版本。包括AMD、Intel、华为、联想在内的五十余家知名IT厂商、独立硬件开发商（IHV）、独立软件开发商（ISV）均参与了此次大会。微软全球资深副总裁、微软中国研发集团主席张亚勤博士在大会上表示：“本届WinHEC大会展示了微软对未来技术趋势的理解，并再次印证了微软在中国打造和完善共赢生态链的决心与承诺。”

开源项目

sourceforge 优秀项目

项目名称：
OrangeHRM
创始人：
Himath Dissanayake，
Ruchira Amarasinghe
何处加入：
http://sourceforge.net/projects/orangehrm/

OrangeHRM是一个人力资源管理（Human Resource Management）系统，它可以帮助企业方便地管理员工信息。OrangeHRM是目前开源社区中最活跃的人力资源管理系统，有着175000次下载，而且OrangeHRM越来越多地被世界各地的公司所认可并使用。OrangeHRM的开发团队使用了一些敏捷开发的实践（XP极限编程），从而为用户提供简单、稳定的系统，并根据用户的反馈对OrangeHRM进行逐步地改进。

安装OrangeHRM需要PHP 5.2和MySQL 5.0.12和Apache Server 1.3或者更新的版本。用户可以下载OrangeHRM与XAMPP一起打包的EXE安装文件进行安装，也可以构建一个*AMP环境，然后通过解压缩并拷贝的方法进行安装。由于OrangeHRM使用PHP开发，所以它并不依赖于某一个操作系统。

OrangeHRM源于创始人对商业HR软件的不满。它们不但在价格方面不适合中小企业，而且商业软件并不能像开源软件那样可以随意定制。因此，OrangeHRM就诞生了。OrangeHRM最初作为一个研究项目切入HRM领域，力图减少企业的成本开销；以开放源代码的方式让更多人可以自行定制这个系统；并以PHP、Apache、MySQL为基础获得更好的移植性。

谈到OrangeHRM的未来，Himath希望它可以成为世界上应用最广泛的HRM系统。

国内优秀开源项目

项目名称：
Amoeba
创始人：
陈思儒
何处加入：
http://sourceforge.net/projects/amoeba

Amoeba是一个使用Java编写的分布式数据库代理，它可以监控SQL执行、数据流量、读写分离，并且可以提供负载均衡、读写分离、数据切分和容错等功能。

目前Amoeba有2个产品：Amoeba for MySQL和Amoeba for Aladdin。这2个产品有不同的适用范围：

1、Amoeba for MySQL：只支持MySQL数据库的代理，需要分析MySQL网络协议，它的代价比Aladdin会小很多，而且性能也较高；

2、Amoeba for Aladdin：是一个使用JDBC驱动的代理，可以使用在所有提供JDBC驱动的数据库上，而且这些数据库可以同时并存于Amoeba for Aladdin后端。

目前Amoeba for Oracle还处于研发状态。其性能也会跟Amoeba for MySQL一样的出色。

就Amoeba for Aladdin这个产品来说，后端的任何数据与Aladdin交互均采用JDBC驱动。而应用程序与Aladdin之间的交互则采用MySQL协议，这个做法主要是想借助MySQL的广泛应用程度以及对各种开发语言的支持。因此对前端的应用来说，Aladdin其实就是一个虚拟的MySQL数据库。

在陈思儒研究MySQL Proxy时，他发现MySQL Proxy并不能非常轻易地解决一些问题（如读写分离、数据切分、水平切分、负载均衡等），而是需要写大量的Lua脚本，这些Lua并不是现成的，而是需要用户自己去写。而这个工作对于并不熟悉MySQL Proxy内置变量和MySQL Protocol的人来说是非常困难的。因此，为了设计一个非常容易使用、可移植性非常强的软件，Amoeba就诞生了。

谈到Amoeba未来，陈思儒说，Amoeba将会更加容易使用、可管理、可动态装载配置以及Amoeba集群等。

创新项目

Screen Anytime是一套运行在Windows平台下，用于监控、管理服务器或工作站屏幕操作细节的视频日志软件。安装后，他可以以连续视频形式自动记录一台计算机上所有登录用户在任何时间的操作细节，包括屏幕显示和鼠标操作。同时，Screen Anytime提供完善的管理和回放机制，使得管理员可以在未来需要的时候，索引或搜索某一时刻的操作录像并回放或导出。

相比于在重要场合普遍安装的安全摄像头监控系统，Screen Anytime则相当于针对重要服务器、重要终端录制计算机操作为内容的安全监控方案。得益于创作人自主研发的SSCV2（Stepok第二代屏幕压缩编解码器）技术，Screen Anytime可以将海量的屏幕操作视频数据压缩到极小空间。

Screen Anytime可适用于以下几个方面：

1、企业重要服务器。一个企业会有多台运行不同重要应用的服务器。每个可以登录这些服务器的管理员，在配置修改服务器时都可能对服务器造成改变。传统的日志系统，只能以文本形式记录大致的步骤。而Screen Anytime可以帮助这些服务器建立更完善的视频日志系统，以作为传统文本日志的有力补充。在出现问题的时候，可以加以参考。

2、政府或企业终端管理。一些企业或政府出于保密需求，或者公司制度需要，要求严格监控员工电脑操作。在不侵犯隐私的前提下，Screen Anytime可以很好地将操作视频进行汇总并管理。此类需求一般包括，涉及机密的政府或军事部门终端；银行ATM终端和柜台计算机；涉及商业机密的企业终端等。

3、客服作息记录。目前通过在线方式提供客户服务的公司已越来越多。如果之前所使用的软件不能对客户的操作进行有效地记录和管理，那么Screen Anytime进行统一地管理将是一个非常好的解决方案。

点评：目前，在安全监控领域，更多的是采用摄像头进行监控。Screen Anytime的出现，提供了一种全新的监控手段。可贵的是，在降低监控成本和记录检索管理方面，Screen Anytime的解决方案有着极强的竞争力。

项目名称：
Screen Anytime
创作人：陈琦

一句话说明：
Screen Anytime是一款PC安全监控软件，可为企业的服务器或工作站终端提供用于安全和管理为目的，且以计算机操作为内容的全天候视频监控日志。
何处体验：
http://www.screen-record.com/screen_anytime.htm

狂雷视频平台是一款整合了视频搜索、视频下载、视频播放、视频转换等全套功能的视频平台。主要采用了Windows网络编程以及UNIX服务器编程等技术。

在狂雷视频平台上，用户无需打开网页，直接在程序里输入需要搜索的关键字即可在狂雷中显示搜索结果（狂雷视频可以对多达746个视频网站进行搜索）。狂雷视频会自动地根据视频的精准度筛选出优质视频资料，并将结果显示在下载窗口，如果你希望看到更多的结果，只需点击“搜索更多”即可。遗憾的是目前版本仅提供按照网站来搜索指定的视频。但其研发人员透露，将在下个版本中添加根据视频清晰度和完整性进行搜索的功能。

狂雷视频的下载功能采用多线程下载与多CDN下载技术，允许用户同时对多个视频进行下载，支持上限为20个。在下载主窗口上，多个下载任务以标签的形式排列，用户可以很方便地查看每个任务的下载进度。

狂雷视频平台还提供了非常实用的视频格式转换功能，方便了喜欢在MP4等手持设备上欣赏视频的用户。点击狂雷程序窗口上的“添加”按钮，只需4步就可以很方便的将下载的FLV视频转换为PC常用的AVI、WMV、RM或是手机等掌上设备支持的MP4、3GP格式，而且可以在右侧调整各种参数设置。

目前，狂雷视频还在和国内著名的手持设备生产厂商合作，在设备出厂时就预装狂雷视频平台。这样，即使是不熟悉电脑操作的用户，也可以方便的获取视频资源，在MP4上欣赏精彩的视频.

点评：现在市面上的视频下载和播放领域，都已有获得用户广泛认可的优秀产品，如迅雷和暴风。狂雷视频平台的优势在于提供了视频的一站式解决方案。另外，方便了用户将视频存储在手持设备或MP4上观看，也是产品的主要亮点。

项目名称：
狂雷视频平台
创作人：CoolSee团队

一句话说明：
狂雷视频平台是一款集视频搜索，视频下载，视频播放，视频转换，视频修复等一系列功能的综合性，专业性视频平台。
何处体验：
www.raydown.com

欢迎推荐优秀创新项目或产品给我们chuangxin@csdn.net

Pocket Cinema

尽管很像，但这并不是一款手机而是一个投影仪。对于投影仪来说，一般人的印象可能是四四方方的造型。而这款手持投影仪不但大小仅有一个手机那么大。而在参数方面，640x480的解析度也可以满足一般的需求了。

Solio

Solio能够在某种程度上解决移动设备充电的问题，它展开后像一只羽毛翅膀一样，只需要在太阳下晒8个小时，就能够提供4个小时的充电时间。重要的是，其大小尺寸只有半只手那么大，与之附加的配件包括Nokia、Motorola、Samsung等手机的转接头以及mini USB接头和充电适配器。

Hymini

天然的无污染能源除了太阳能之外，风能也是其中之一。Hymini就是这样一款利用风能的手持发电器，同样也配备了各类移动设备的转接口。除了通过风力发电之外，还可以外接太阳能板。对于喜爱慢跑、户外或者骑单车的朋友，这是另外一种获取环保能源的方式。

The Pimped Out John

根据调查，一个人一生中在卫生间中花去的时间大约有11862个小时。不必说，在这段期间内需要找一些有意思的事情做。这套组合就是Roto-Rooter的解决方案——The Pimped Out John。它包括20英寸的平板显示器、Xbox 360、星球大战的DVD、笔记本电脑、iPod、杂志架、自行车训练器材、电子加热/制冷设备等等。

WALL-E

机器人已经不再仅仅属于狂热分子，很多公司都推出了商业化的可编程机器人玩具。现在Disney也开始进入这个市场。它根据自己电影中的角色创造出了耳熟能详的机器人角色。例如左边这个就是在《机器人总动员》中的角色WALL-E。

Novint Falcon

Novint Falcon是一种全新的PC游戏输入设备，准确地说是一个带枪柄的手持瞄准器。它可以在游戏中通过力反馈让玩家得到更多的游戏场景的反馈，在这一点上与传统意义上的鼠标是完全不同的，你甚至可以感受到每种武器不同的后坐力。游戏公司Valve也宣布在未来的几款游戏中将增加对此设备的支持。

编译领域里程碑之作

——龙书《编译原理》

■ 文/孟岩

编译技术一向被认为是软件开发这个以工程和实践为主的学科当中真正的“火箭科技”。一名程序员，当他掌握了基本的编程技能，并且具备一些实践经验以后，如果要求在编程技术上进一步深造，则无非有三个方向：其一是钻研高超的算法；其二是深入计算机和网络体系结构；其三则是提升编程的抽象层次，三条路线彼此相关，无分优劣。不过从编程技术本身来说，提升程序的抽象层次具有化繁为简、四两拨千斤的妙趣，尤其引人入胜。六十年来计算机和软件技术的一切发展，就是抽象层次不断提升的过程。从最早的硬件开关、机器语言、顺序加跳转的指令流，到后来的高级语言、结构化编程，再到近二十年来相继兴起的面向对象、构件技术和面向服务架构，无不遵循着这一逻辑。提升抽象层次有多种手段，函数库、类库、框架、构件、服务，不一而足。而在所有的手段当中，计算机语言是层次最高、能力最强、最富妙趣的一种，别的手段能够达到的效果，语言都可以更轻松地实现，而语言所具备的某些效果，则没有别的手段可以企及，语言是提升软件抽象层次的终极手段。这并不奇怪，人类的抽象能力最终归于语言，这是在哲学范畴内已经被深入讨论过的命题。因此，计算机语言的理论及其处理，被很多一流学者认为是计算机科学中最成熟、最优雅、与实践结合最完美的部分，其重要性不言而喻。

在实践中，如果一个程序员具备强悍的计算机语言处理能力，则他的程序编写和设计能力将会比其他人有层次上的超越，往往能够做出创新性的成果，给开发的效率和效果带来本质性的提升。

放下计算机语言理论不提，若谈及编译技术，这本由几位著名计算机科学家合著的“龙书”自出版以来就称为本领域的经典著作，被世界众多著名大学选作编译技术本科和研究生课程的教材，可以说是编译技术的“圣经”。也正是因为这本书的名气最大，围绕它的争议也最多：赞誉它的人说它名门正派，体系完整清晰，技术点交代一丝不苟，语言严谨，思路缜密，义正词严，绝无投机取巧之媚态，凡认真学习编译技术者必读之。而批评者则认为，面对编译技术这样一个枯燥艰深的课题，本书严谨有余，活泼不足，显得太刻板。然而更多的人还是认为，尽管这本书不可避免地有自己的缺点，但是全面衡量之下，这本书仍然不愧是编译技术领域最全面最重要的著作，是编译领域的“THE BOOK”。在我看来，“龙书”是一本严谨系统的学术性经典教材，对于有意系统了解编译技术的读者来说，确实不愧为必读之书。

本书第一版于1986年出版，横行天下二十年，可以说已经是个奇迹了。2006年，原书三位作者与一位新作者合作，根据二十年来编译技术的发展，编写了本书第二版。新版风格与特色与老版相似，但充分反映了该领域最新的重要成果，并且采用Java作为范例语言，体现出了作者开放进取的心态。更为重要的是，“龙书”作者认识到，大多数读者不会亲自去构造一个编译器，而是希望从编译这个思想技术宝库中汲取营养，因此特别注意了概念思路的叙述和内容的启发性。全书有两大重点，前半部分侧重于传统的词法和文法分析，后半部分则详细介绍优化原理和技术。这两个重点，可以说是编译技术思想库中最璀璨的宝贝，在很多领域都有着广泛的应用和启发性。可见作者绝不是就事论事，狭隘执着于编译本身，而是希望尽可能给读者提供更具实践意义的工具和锻炼。

由于出现了Lex/Yacc，特别是近年来流行的ANTLR等LL(*)语言处理程序生成工具，如今编写语言处理工具的难度已经大大降低了。然而若想达到较高的境界，系统的学习和实践本书的内容仍然是必要的。相信新版“龙书”中文版的出世能够帮助更多年轻程序员真正理解编译技术，并且为我所用，做出创新的成果。■

成功演示六要素

——《演说之禅：职场必知的幻灯片秘技》节选

■ 文/Garr Reynolds

随着微软Office办公软件一统天下，可以说我们进入了一个幻灯片的时代。从公司的会议室到大学的教室，几乎随处可见在幻灯片前侃侃而谈的演讲人。幻灯片越来越成为我们工作生活的一部分。但随着人们对幻灯片的依赖不断加大，人们也有了新的烦恼。每个人都在为如何使自己的幻灯片更绚、更酷而烦恼。

在准备演说时，切记要离开电脑。许多人犯的一个根本性的错误就是，大部分的时间坐在电脑前面，琢磨着怎么讲，讲什么。其实在设计之前，首先要做的是对演说全盘考量，并确定主旨，只有心绪平静下来才可能都到。在不要电脑干扰的情况下，你独自一人或集体一同寻求灵感。你不时地停下来纵览全局，明确核心思想。虽然有些细节问题还有待充实，但你对演说的内容已更加地清晰和明确。下一步要做的，就是为核心思想、事实以及理论依据组织一个逻辑架构。这个逻辑框架会将演说内容安排地有序得当，令你在演示时更加流畅，同时也更易于观众理解。

在你将纸上的各种想法或观点以PowerPoint或Keynote呈现出来之前，你需要考虑如何让观众对演说产生共鸣的问题。是什么使你的演说充满智慧，令人难忘？如果你有志于做一场令人印象深刻的演说，那么必须时刻考虑如何巧妙地安排所讲内容，使其令人难忘。

大多数著名的有关演说技能的书籍并不讨论演说本身，也不会讨论如何使用幻灯片软件。齐普•希思（Chip Heath）和丹•希思（Dan Heath）兄弟俩的《为何有些想法吸引人，而有些则不行》（Made to Stick）一书就是其中之一。希思兄弟感兴趣的是，为何有些做法能吸引人，令人印象深刻，而有些则行不通。对于此问题，二人总结出了演说六原则：简单（Simplicity）、意外（Unexpectedness）、具体（Concreteness）、可信（Credibility）、情感（Emotions）和故事（Story）。没错，这些英文单词的首字母刚好能拼成SUCCESS（成功）。

在演说中（包括幻灯片演说）遵循上述六原则是很容易的，但大多数人却做不到，这是为什么呢？作者把其中最大的原因归结为“知识祸根”。所谓的“知识祸根”是指演说者在讲述的过程中没有考虑一个外行人听自己的演说会有怎样的感受。他可以用抽象的语言对所讲题目高谈阔论。至于讲了些什么，他自己认为是显而易见的，但却不知道不具有相关知识背景的听众理解起来会有多困难。六大原则——SUCCESS—就可以避免“知识祸根”，从而使自己所讲的容易让人记住。

这里有个例子，可以说明留存持久、有力的言语和枯燥无力的言语之间的区别。以下两句话意思相同，其中一句你肯定很熟悉。

“我们的任务是通过团队革新和航天战略计划部署成为世界太空业的先驱。”

“……把人送上月球并在十年后安全返回地面。”

第一句话听起来像是当今某个CEO说的，理解起来都很困难，更别说使人印象深刻了。第二句话其实是J.F.肯尼迪在1961年做某个演讲时说的，体现了上述六原则的应用。这句话激励了一个民族为航天事业的发展而努力，从而改变了整个世界。肯尼迪，或者说至少是他的讲稿撰写人，非常清楚，抽象的话语是无法令人印象深

刻，受到激励的。但是现今又有多少CEO开口不是“使股东的利益最大化”之类的话呢？下面是对《为何有些想法吸引人，而有些则不行》一书中提到的六原则的总结小结。在我们阐述想法或构思包括演说、演讲等交流形式时，需要特别记住这六大原则。

简单

如果每件事都是重要的，那就没什么是重要的了；如果每件事都须优先考虑，那就没什么优先权可言了。你要毫不留情地简化你的信息（不是过度简化），努力做到去其浮华留其精髓。我不是指那些电视和电台播出的老套的新闻语言。如果你尽力了，任何信息都可以以最精炼的语言表达。你的演说想说明什么？实质是什么？其意义又何在？这些便是准备演说时所应考虑的。

意外

你可以通过使别人感到意外而引起他们的兴趣。让别人惊讶一下，那会令他们兴趣尤生。但要维持这种兴趣，你需不断激发他们的好奇心。最好的办法就是向他们提问或使其生惑，再解惑。使听众意识到他们知识上的一个漏洞，然后通过向他们提供解答或加以引导，从而填补这个知识漏洞，这就是解惑。这就好比你带领听众开始了一个探索之旅。

具体

不要使用抽象化的概念，多举些实例使说话内容具体化。希思兄弟指出，谚语就是个不错的选择，它能把抽象的概念具体化、简单化，同时有说服力、令人难忘。比如说，“一石二鸟”这个谚语是不是要比“通过提高各部门的工作效率使生产力达到最大化”更加简化却更有说服力呢？肯尼迪的那句“送人去月球再返回”的话也是如此。具体实在的事物容易使人联想和锁定其意含。

可信

如果你在某研究领域有所作为且很出名的话，你就拥有内在的可信度（但现在看来越来越不是这么回事）。但是多数人并没有那种可信度，为了证明某一说法，就必须以事实说话。比如，为了证明我们是市场的领头羊，那么就要列举出客观的数据来证明。希思兄弟说，单纯的统计数据并没多大意义，关键还要看其背景和内在涵

义。说话时要多用一些人们容易联想到的事物。比如，“五小时的电池时间”和“拥有足够的电量，使你从旧金山飞往纽约的途中用iPod无间断地观看你最喜欢的电视节目”，这两种说法哪个更具可信度呢？建立可信度的方法有许多，比如引用某客户或媒体的话语等。相反，长篇地赘述公司的历史只会使听众生厌。

情感

人是情感动物。仅仅向观众放一遍幻灯片是远远不够的，你还必须唤起他们的内心感受。使观众真切感受所讲内容的方法非常多，图片的使用就是其中之一。图片不仅能够帮助观众更好地理解所讲要点，还能触动他们的内心，激发其对所讲内容情绪体验。比如当介绍美国卡特利亚飓风和洪水泛滥所造成的严重后果时，你可以陈列要点、数据，但是，记录灾后事发地混乱情况以及人们痛苦表情的照片所取得的效果是文字、数据所无法企及的。一提到“卡特里娜飓风”的字眼，人们的脑海里就会产生一系列灾难性的画面，栩栩如生。人是与人存在着情感维系，而不是与抽象的字符。如果可能，尽可量使表达具有人性化的色彩。比如，“一百克脂肪”对你来说是个很具体的概念，但对其他人来说可能很抽象。一张包括一大包薯条、两个汉堡包和一大杯巧克力奶昔的图片就能帮助人们理解“一百克脂肪”的概念，从而唤起他们的情感体验。

故事

我们无时不刻都在讲述故事，与人进行交流。除了使用语言讲述故事外，我们还可以通过艺术或音乐的形式与人沟通。在一教一学中，故事伴我们成长。在日本，新员工刚入厂，往往都会由老员工带，从老员工那了解，公司的历史，文化以及如何工作等等。老员工们使用的就是讲述故事的方法。拿戴头盔为例，师傅们会说某某某在工地没带头盔而发生了惨剧，这样徒弟们一下子就记住了。每次去工地时都会记得带上头盔。相比死板的条文条规，故事则更能引起人们的注意，并容易牢记。好莱坞、宝莱坞大片也好，独立电影也罢，为什么人们这么喜欢看电影呢？还不是因为其中吸引人的故事情节吗？那为什么轮到聪明能干的故事狂热者做演说的时候，他们不使用故事、实例或插图去论述问题，而偏偏选择晦涩难懂的话语呢？伟大的创意和精彩的演说必然包含故事。如今杰出的演说者常常通过亲身经历去表达观点和看法。向别人解释一个较复杂的想法时，最简单的办法就是运用具体事例或故事进行论述。故事往往能够使听众留下更深的印象。如果你希望他们记住你的内容，不妨多用一些有趣、精炼的好故事或事例来强化你的核心思想，从而在他们的脑海里留下更深的印记。■

（本文章节选自电子工业出版社已出版的《演说之禅：职场必知的幻灯片秘技》一书）

新书上架

Oracle Database 10g性能调整与优化

作者：（美）Richard J. Niemiec

出版社：清华大学出版社

该书作者Richard J. Niemiec，是全球仅有的6位Oracle认证大师之一，为独立Oracle用户组(IOUG)的前任主管以及中西部Oracle用户组的现任主管。书中内容可以有效地帮助初学者和Oracle专业人员理解并更好地优化Oracle系统。该书提供了监控、分析和优化Oracle Database 10g的方法，并包含了详细的案例研究、最佳实践和丰富的Oracle新的调整特性的代码示例。基于作者的多年经验和实践，还对多种应用提供了很多实际的技巧。书中设置有包含如下领域的大量信息和技巧：网格控制、ASH、AWR、ADDM、块级调整和数学性能建模，可以帮助专业人员进行进行系统性的提升。无论对于初学者，还是富有经验的DBA，相信都会从此书中找到有价值的内容。

Windows程序设计（第5版•英文版）（上下册）

作者：Charles Petzold

出版社：人民邮电出版社

本书作者 Petzold是享誉世界的微软技术大师、Windows先驱奖得主，1994年时，作为仅有的7人之一被《Window Magazine》和Microsoft公司授予Windows Pioneer奖，以表彰他对Microsoft Windows的成功做出的贡献。很多有经验的Windows程序员都知道，如果在工作中遇到了技术难题，最好的解决办法就是去查Petzold的书。此书正是作者经典之作，书中内容博大精深，阐述透彻而流畅，是一部Windows程序设计的圣经，一部畅销十年不衰的佳作！

作者根据最新的Windows操作系统权威技术修订了旧版，再次演示了基本的Win32程序设计的API核心内容。书中内容包括：Windows基础知识、Unicode概述、图形、内核和打印机、声音和音乐、动态链接库、多任务和多线程等。书中采用代表性示例，为使用 Windows 95、Windows 98或 Windows NT的各级windows程序员提供了最基本的参考和指导，便于学习和检索。

软件架构的艺术

作者：李伟 吴庆海

出版社：电子工业出版社

架构之美，在于和谐。作为《架构之美》丛书之一，《软件架构的艺术》聚焦于软件架构行业，全面介绍软件应用系统架构的基本原理、方法以及经典的实践经验，并创新性地提出为确保高质量的架构设计而应该遵循的“架构生命周期ArchitectureLifecycle”理论。

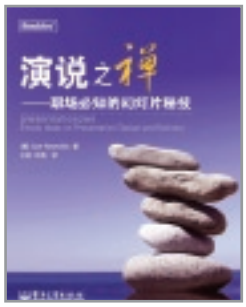
本书涵盖了软件架构生命周期中涉及的创建、评审、执行、演化恢复等重要活动、流程及实践经验，系统性地介绍了关于软件架构方面的基本概念、体系及方法。希望能够给相关领域从业人员以帮助，缩短其学习和成长的周期。

演说之禅——职场必知的幻灯片秘技

作者：（美）雷纳德（Reynolds,G.）

译者：王佑，汪亮

出版社：电子工业出版社



时近年末，各种大小会议蜂拥而来，准备一个精彩的PPT、做一场让人印象深刻的演讲也许成了你展示自己和产品绝好机会。这本书的推出，让很多人眼前一亮，本刊虽然已在前文安排了节选，但这里仍然再次推荐，因为它确实是一本实用的佳作。

作者加尔•雷纳德是一位幻灯片设计师，也是一位全球知名的交流专家。他用朴素的道理、生动的案例阐述了幻灯片设计的基本理念和根本原则。这些简单易行的原则和方法，可以帮助职场人士以一个独特、简便、直观、自然和针对性的角度看待幻灯片演示，进而从设计和演示幻灯片的苦海中解脱出来。特别是作者把禅宗秉持简约的原则与PPT制作演示相互关联，使得我们能够从思想之高峰看待如何制作PPT。

如何立竿见影地做好PPT？一，幻灯片上的内容不应是你叙述要点的重复，而应起到一个推波助澜的作用。二，不用使用质量差、低俗的图片。三，不要使用渐变、旋转或其他幻灯片切换效果。

2008年度CSDN 10大博客之“绝代双雄”

走出软件作坊

作者：阿朱（吕建伟）

出版社：电子工业出版社

2008年初，有一个博客在CSDN上异常火爆，博主阿朱以“三五个人十来条枪，如何成为开发正规军”的系列文章，总结了自己从业十年来在技术项目和技术团队管理方面的经验和思考。年底，这些倍受欢迎的文章以《走出软件作坊》一书的形式，完整地呈现在了读者面前。

CSDN、《程序员》杂志总编韩磊专门作序推荐。他认为：这本书的价值，一是为作者十年实践所得，书中不虚谈理论，全部心得皆来源于其实践所得，对于本土小型或创业型软件企业，具有宝贵的参考和借鉴价值。其次，书中所体现的实践知行观，体现的是一种不盲从的反思精神，这是软件企业和软件从业者最该思考的。每家公司都有着独特的环境和氛围，所谓管理、规范、制度、方法和人情，缺一不可但又必须相辅相成。

这本从阿朱7年职业经理人心得总结而来的《走出软件作坊》，可以称之为一本中小IT企业和创业团队的实战管理手册。书中形式活泼，内容独特，以作者多年宝贵经验，完整讲述三五个人十条枪成为软件开发正规军的发展之路，为很多创业团队提供了解决发展过程中常见问题的应变之道，同时精辟地论述IT企业项目管理和团队建设的精髓，内容涵盖组织结构、团队文化、软件过程管理、团队激励、绩效考核、职业发展规划等，具有极强的现实指导意义。

疯狂的程序员

作者：绝影

出版社：人民邮电出版社

绝影在CSDN Blog上连载《疯狂的程序员》，从2007年12月22日开始，到2008年9月15日全文完成，9个月时间写了35万字。甚至在四川地震后，身处灾区的绝影，依然在帐篷外坚持写作。正如绝影所说，一开始，他并没有想到会出书，只是因为CSDN网友的热情和支持，让他一直坚持了下来，至于后来的出版，只能算是一个意外的收获。

这本《疯狂的程序员》，是一部真实再现程序员成长历程的原创小说，作者以他学习、工作、生活为原型，记述了大学、工作、创业三部分历程，内容精彩迭出，其中作为主线的实际项目案例都是基于作者深厚的技术积淀。小说不但描写了软件行业中形形色色的人和事，而且故事和语言形象而生动，充满了智慧的职场色彩。

CSDN总经理蒋涛评价说，“我们在IT图书市场看到的，要么是IT成功人士包装后的传记，要么是实打实的技术书籍。实际上在软件开发界并不缺乏有趣的人物和精彩的故事。我一直比较喜欢技术人文的书籍，就是把技术表现得不那么硬梆梆，很高兴在CSDN Blog上产生了这样一部真实的程序员故事，从读书到工作，从打工到创业，绝影的成长十分真实，因为真实而精彩，因为真实而有价值。”

《疯狂的程序员》也给更多CSDN网友树立了典范：程序员也能写书，而且能写出引起大家共鸣的优秀作品。

全球图书销售排行榜

	Amazon	第二书店	天龙书局（台湾）
1	The Digital Photography Book	大象——Thinking in UML	Learning jQuery 中文版
2	The Black Swan: The Impact of the Highly Improbable	编译原理（原书第2版）（龙书）	Linux 装置驱动程序之开发祥解
3	YOU: Being Beautiful: The Owner's Manual to Inner and Outer Beauty	疯狂的程序员	程序之美—微软技术面试心得
4	TBeginning iPhone Development: Exploring the iPhone SDK	Erlang 程序设计	ASP.NET 3.5 应用系统专题实作
5	Presentation Zen: Simple Ideas on Presentation Design and Delivery (Voices That Matter)	走出软件作坊	Google APIs 程序工具锦集
6	The Kindle Cookbook: How To Do Everything the Manual Doesn't Tell You	Windows 核心编程（第5版）	Web CSS 网页设计大全
7	The Unofficial LEGO MINDSTORMS NXT Inventor's Guide	天书夜读：从汇编语言到 Windows 内核编程	深入浅出 C# (Head First C#)
8	iPhone: The Missing Manual: Covers the iPhone 3G	编程珠玑（第2版）	现代嵌入式系统开发专案实务-菜鸟成长日志与专案经理的私房菜
9	Fallout 3: Prima Official Game Guide (Prima Official Game Guides)	代码之美 Beautiful Code（中文版）	Short Coding写出简洁好程序－短码达人的心得技法
10	Mac OS X Leopard: The Missing Manual	算法导论（原书第2版）	Flex 3 彻底研究 (Adobe Flex 3: Training from the Source)

蒋清野，1999年获得清华大学学士学位，2000年获得美国伊利诺伊大学（University of Illinois at Urbana-Champaign）硕士学位，现任Sun中国技术社区高级经理。

回顾：OpenSolaris 2008.11

■ 文 / 蒋清野

OpenSolaris 2008.11最终于12月1日正式发布。尽管只是晚了一天，依然足以证明用日期来作为软件的版本号是一件不甚靠谱的事情。众所周知，在这最后两个星期里Sun公司宣布了裁员6000人的决定。值得庆幸的是，这个决定并没有影响到Sun公司坚持走开源路线的决心。

2008.11版本的OpenSolaris在改进用户体验方面颇费心思。使用过2008.05版本的用户大都体验过pkg命令崩溃或者是包管理器（Package Manager）假死等等令人不快的情景，在新的版本中这两个功能都得到了显著的增强。在此基础上2008.11版本增加了更新通知功能，使得用户可以及时地通过网络下载安装最新版本的软件。软件资料库pkg.opensolaris.org设置了release, contrib, dev三个不同的目录来区分正式发行版本、社区贡献版本和开发中版本的软件，使得社区成员向OpenSolaris项目贡献打包好的软件成为可能。Sun公司也开通了官方的软件资料库pkg.sun.com，其中extra目录提供由于授权问题无法自由地重新发布的软件，support目录则向Sun公司的订阅用户提供支持。不足之处是Sun公司依然没有发布制作IPS服务器镜像的方法，因此全球的OpenSolaris用户都要通过位于美国的IPS服务器下载和安装应用程序。（唯一的例外是中国，国内用户可以使用Unix-Center.Net设立的IPS服务器镜像。）

在Gnome 2.24, FireFox 3, ThunderBird 2和OpenOffice.org 3的基础上，2008.11版本提供了一个相对可用的开发者桌面。你可以用SongBird来听MP3，用Tracker来进行桌面搜索，用Pigdin来和MSN或者是QQ上的朋友聊天，用Gobby来进行远程协作，用Transmission来完成BT下载，用Sun Studio、NetBeans或者是Eclipse来开发应用程序。只需要安装一个叫做webstack的软件包，即使是从来都没有做过Web开发的新手也可以轻松搞定Apache、MySQL和PHP的安装和配置。更重要的是，利用OpenSolaris所特有的DTrace特性，开发人员可以轻松地定位和消除应用程序中的瓶颈。即使是针对Linux操作系统开发的应用程序，也可以在OpenSolaris上利用DTrace进行性能调优，然后再部署到Linux操作系统上。

2008.11版本中最引人注目的特性，毫无疑问是Time Slider。简单地讲，通过拖拽文件管理器上一个代表时间的滑快，你可以看到同一目录在不同时间的内容。相信大部分用户都有过误删文件的经历，这时候我们总是想“如果我能够回到过去就好了”。利用Time Slider，你可以放心地编辑和删除您的任何文件或者是目录，因为在任意时刻你总是能够回到过去。这看起来像是一个超级的回收站，也有朋友将其和Mac OS上的Time Machine相比较。其实Time Slider所利用的是ZFS的特性，通过定时地制作指定目录的快照来进行数据备份。ZFS的快照是增量式的，它所保存的不是文件本身而是文件改变的增量，因此它不会显著地增加对存储的需求，也不会显著地影响操作系统的性能。

Solaris操作系统的忠实用户可能会怀念Flash Install，它使得系统管理员能够轻易地同时安装和配置千百台相同或者是相似的Solaris服务器。从2008.11版本开始，OpenSolaris尝试着提供一个全新的自动安装技术。和Solaris操作系统相类似，OpenSolaris系统管理员可以配置一台安装服务器，不同配置的待安装机器可以访问该服务器，获得符合本机配置的安装清单，从而自动地完成安装和配置。此外，

打算自己制作OpenSolaris发行版的用户可以利用Distribution Constructor来定制化OpenSolaris发行版。这个工具接受一个安装清单文件作为输入，输出一个可启动的OpenSolaris安装光盘映像。

Linux操作系统的用户可能会问："那又怎么样？你说的这些Linux大部分都有，并且比OpenSolaris要做得更好。OpenSolaris到底有独特的优势？"

Sun公司的忠实粉丝会一遍又一遍地回答："ZFS、DTrace、SMF、ZFS、DTrace、SMF。"

然而我认为这不是OpenSolaris的本质。

近些年来，计算机的性能不断提升而其成本不断降低，使得越来越多的人涌入计算机相关产业。仅以中国为例，1995年中国设置有计算机相关专业的大专院校不过100所左右，到2008年这个数字已经突破了800所。从业人员的增加必然要求相关行业降低其进入门槛，对于软件产品来讲就是在降低获得成本的同时提高用户体验。换句话说，就是软件要越来越便宜并且越来越容易使用。在过去的10年间，Solaris操作系统从收费软件变成了免费软件，但是其用户界面并没有发生根本性的变化。在同一时期，Windows坚持收费的路线但是不断改善用户体验，最终成为普通用户首选的桌面环境。开放源代码的Linux操作系统由于免费而吸引了大量需要Unix操作系统的开发人员，这些开发人员对Linux的贡献又使其性能和用户体验逐渐接近甚至超越了传统的Unix操作系统。在Linux咄咄逼人的攻势之前，Novell转向了Linux，SCO倒下了，IBM和HP在拥抱Linux的同时将AIX和HP-UX定位为大型机操作系统。作为硕果仅存的一个仍在不断发展的商业Unix操作系统，Solaris的其市场份额也在不断萎缩。

OpenSolaris的本质，是放弃旧的开发与销售模式，拥抱新的开发与销售模式，从之前的"技术决定市场"转向将来的"市场决定技术"。说得更简单点，就是放弃学院派的自我崇拜，倾听用户的反馈，切实改进用户体验，按照用户的使用习惯提供用户需要的功能。

这就是为什么Sun公司要收购MySQL。这也是为什么OpenSolaris要支持Eclipse。使用Java语言的开发人员一定非常熟悉NetBeans与Eclipse之间的明争暗斗，不过最近两年来Sun公司对于Eclipse的态度已经大为缓和。举个例子说，在即将发布的JavaFX SDK 1.0中，很有可能会提供一个Eclipse的插件，使得开发人员可以使用Eclipse开发JavaFX应用。在未来一到两年里，会不会有更多的NetBeans特性被贡献给Ecipse基金会，甚至是出现一个基于Eclipse的NetBeans？尽管存在众多的不确定性，我相信现在说还不为时过早。

Linux用户更为关心的其实是另外一个问题：OpenSolaris什么时候会使用GPL授权协议发布？Linux用户什么时候能够用上ZFS、DTrace和SMF？

这个问题也许可以转换成下面一种表述：Sun公司如果希望其发布的某种操作系统得到Linux用户的青睐，那么这个操作系统需要具备如下特点：

（1）支持Linux所支持的硬件

（2）使用Linux所使用的桌面

（3）采纳Linux用户的使用习惯作为系统缺省设置（例如bash）

（4）使用GPL授权协议

（5）提供ZFS、DTrace、SMF以及未来Sun公司的其他技术创新

在技术层面上，广泛的硬件支持是OpenSolaris操作系统最大的难点。虽然Sun公司有一个庞大的团队在从事x86相关硬件驱动的开发，为OpenSolaris贡献驱动和补丁的社区人员也与日俱增，然而要赶上Linux对硬件支持的水平还需要不短的时间。Linux的硬件驱动程序虽然是开放源代码的，但是由于授权协议互不兼容的原因，OpenSolaris开发人员完全无法利用这些开源社区已有的成果对自身进行改进。因此，使用GPL授权协议重新发布OpenSolaris源代码和二进制代码，也许是一种双赢的解决方案。

在非技术层面上，OpenSolaris作为Linux的竞争对手，在情感上难以获得Linux用户的支持。2005年Sun公司使用CDDL授权协议发布OpenSolaris项目的时候，曾经将CDDL授权协议与GPL等授权协议进行比较并对GPL授权协议做出了一些负面的评价，引起了Linux用户的普遍反感。Linus Tolvalds更是指出："由此看来，Sun公司只是打算利用Linux社区的资源（尤其是驱动），而不是真的想要回馈开源社区，尤其是ZFS文件系统。"近几年来，Sun公司内部对GPL授权协议的态度发生了重大转变，譬如2006年11月开放Java虚拟机源代码时采用了GPLv2授权协议，2008年9月开放xVM源代码时采用了GPLv3授权协议。最近，Sun公司首席执行长官Jonathan Schwat也曾经多次在公开和非公开场合表示Sun公司不排除使用GPL授权协议发布OpenSolaris的可能性。这种可能性一旦成为现实，就犹如柏林墙之倒掉，昔日的宿敌终于成为朋友。往大的方面讲，OpenSolaris与Linux两个开源社区冰释前嫌，才有可能激发如上所述之技术融合，最大程度地实现开放源代码技术的价值。往小的方面讲，Sun公司消除了Linux社区的敌意，才能够更有效地宣传和推广其OpenSolaris操作系统。

因此，不管是在技术层面还是非技术层面上，使用GPL授权协议发布OpenSolaris项目对于Sun公司和整个开源社区来说都是个双赢的选择。

这件事情是否真的会发生？没有人能够预测未来。然而思想的桎梏一旦被打破，一切皆有可能。■

■ 责任编辑：李雨来（liyl@csdn.net）

张永强
1996年毕业于四川大学，十余年的研发经验，长期关注软件相关技术发展。现任杭州华视数字技术有限公司技术总监，从事数字电视相关产品的设计与研发工作。

Java与数字电视：天堂还是地狱？

■ 文 / 张永强

Java的嵌入式源流：历史其实是这样的

回到Java诞生的20世纪最后一个充满希望和机会的十年，Sun公司在1991年初成立了Green Team，目的是要发展一种新架构，而这种架构必须能够在消费性电子产品作业平台上运行，经过了17个月的研发诞生了Oak这个Java的前身。绿色小组在1992年的9月3号，发布了一款由Java技术之父James Gosling领军研发，名叫Star Seven(*7)的一部交互式的掌上型家用娱乐装置，可透过使用动画触碰式屏幕的使用者接口来控制其他电子设备。

1992年11月，Green Team被转化成了“FirstPerson有限公司”，一个Sun公司的全资子公司，团队也被重新安排到了硅谷的帕洛阿尔托。FirstPerson团队对建造一种高度互动的设备感兴趣，当时代华纳发布了一个关于电视机顶盒的征求提议书时（Request for proposal），FirstPerson改变了他们的目标，作为对征求意见书的响应，提出了一个机顶盒平台的提议。但是有线电视业界觉得FirstPerson的平台给予用户过多的控制权，因此FirstPerson的投标败给了SGI。与3DO公司的另外一笔关于机顶盒的交易也没有成功，由于他们的平台不能在电视工业产生任何效益，公司被并回Sun公司。

项目虽然完成了，Green Team却没有在智能家电上获得成功。

随着Internet的前进步伐，工业界对适合在网络异构环境下使用的语言有一种非常急迫的需求，在Sun的首席工程师Bill Joy等大牛的推动下，James Gosling改变了Oak和绿色计划的方向，Java在1995年的3月23日诞生了。接下来的事情就是众所周知的了，Java迅速成为Web服务器端的主流。

Java在嵌入式领域经过10年的发展的，经过了Personal Java，KJava，Java在1999年被Sun切割成J2ME，J2SE，J2EE三个不同的架构，并在工业界发展成为主流开发语言，在J2ME架构上，形成了若干标准，如CDC/CLDC，MIDP等等。应用的领域也逐渐在手机，PDA等消费电子产品市场上扩展。目前几乎所有的智能手机和PDA上都支持Java。但是要看到Java在嵌入式领域的应用才是Java产生的初衷，Java在嵌入式领域尤其是个人信息终端，如智能手机和PDA等的消费电子产品领域还算是成功的。

Java与中国数字电视：国际标准与中国国情?

远在2000年，“国际数字视频广播”组织(DVB-Digital Video Broadcasting，DVB)就已经宣布选用Java作为交互式电视的基础技术架构，并在2001年发布了基于Java的互动电视标准——数字视频广播多媒体家用平台“DVB-MHP(DVB-Multimedia Home Platform)” 标准规范1.0。该标准是许多数字电视中间件的标准的来源，包括到目前在漫长的7年间一直处于制定中的中国的数字电视中间件标准DTVM。

中国数字电视中间件国家标准DTVM是构建在Java之上的，提供数字电视交互控制的接口，隔离硬件方案的移植性，从技术角度来看，是绝对可行的道路，而且是现实的选择。2001年中国数字电视中间件标准组在信息产业部标准化研究所成立，参与单位都是业内的主要公

司，到目前为止，推出过3个正式的草案，却都没有在最后形成正式颁布的国家标准。

Java在嵌入式领域的应用，Sun的知识产权是一个绕不过的问题，按照Sun在嵌入式领域的操作惯例，授权费用是肯定要给Sun公司提交的，而且相比较机顶盒目前成本在60美元的情况下，授权费用放在2美元是很高的，考虑到中国数字电视机顶盒的巨大市场容量，在2006年，中国数字电视中间件标准组在信息产业部的组织下，代表中国数字电视产业联盟和Sun公司就Java的知识产权进行了谈判，Sun公司在授权费用上进行了比较大的让步，结果是每机顶盒不超过0.8美元。即使这样也没有在推动Java在数字电视的应用上起到什么作用。

Java的数字电视麦城：理想与现实的差距

在数字电视产业内部有很多占有巨大市场优势的公司，其中嵌入式浏览器的厂家因为市场操作的原因，刻意混淆中间件的概念，在数字电视市场上，大力宣扬浏览器就是中间件。而当前在单向数字电视市场上浏览器应用因为先入为主，目前占据市场上的主流应用，给基于Java的中间件解决方案的推广带来了相当大的难度。

目前在中国数字电视产业内部，基于Java的中间件解决方案在中国数字电视产业的发展不能不提到上海蓝信软件技术有限公司。蓝信是当前在数字电视硬件方案业内执牛耳地位的意法半导体（ST）的控股公司，ST作为业内最大的硬件解决方案提供商，认为数字电视中间件是中国数字电视机顶盒解决方案的大势所趋。结果在国家标准一再难产的情况下，专业的中间件软件提供商的前景渺茫，怎么也敌不过国情这个关口，所谓人算不如天算，虽然蓝信做出了一定的成绩。借用ST的市场能力，2006年内蒙古自治区在呼和浩特的单向平移机顶盒全面采用蓝信的中间件解决方案。在国内，这个案例是基于Java的解决方案的唯一全网成功案例。之后在数字电视中间件国家标准一再难产的情况下，在2007年甚至蓝信在数字电视中间件上的技术带头人都离开了蓝信。Java在中国数字电视领域的应用真的是走了麦城。

Java在国内数字电视领域的前景：前路茫茫

目前模拟电视向数字电视的平移已经基本过半，受平移型机顶盒的成本所限和既有应用的影响，Java中间件在平移型机顶盒上的应用可以说基本上没什么希望。而目前国内数字电视市场上，深入挖掘单向数字电视潜力，使用既有机顶盒的能力进行增值业务的运营是产业热点。在这种情况下，使用嵌入式浏览器作为单向数字电视应用的核心是目前能够看到的解决方案。在可预见的将来，单向数字电视机顶盒在基础架构上转向Java方案已经是不太可能的一件事情。

从消费电子类产品如手机解决方案来看，RIM的Black Berry，Google的Android等系统来看，都使用了所谓非Java的类Java解决方案，也就是说不是经过Sun认证的Java方案，使用类Java语言，绕过Sun的控制。是否在数字电视机顶盒上也可以使用这样的手段来绕开Sun的授权费用？甚至在机顶盒上直接使用Google的Android平台？

在双向数字电视机顶盒上Java解决方案是否可以进入市场？现在情况还不明朗，决定是否使用Java方案的原因目前看主要是应用模式，从目前双向数字电视应用还是以VOD作为市场突破口的情况下，Java方案的前景不乐观。

Java与数字电视，到底是天堂还是地狱？■

■ 责任编辑：李雨来（liyl@csdn.net）